File Edit View History Bookmarks Tools Help

Multiple tables

www.pgrocer.net/Cis50/multiple.html

Relating multiple tables using SQL

For this handout, I am using three tables: donor, donation, drive. Donor and donation can be linked/joined based on idno and donation and drive can be linked/joined based on driveno. There is no direct connection between donor and drive. The rows of data and description of these three tables is shown below:BR>

SQL CODE: DONOR Table

Donor database includes the three tables shown.

```
SQL> SELECT * from donor;

IDNO  NAME            STADR           CITY       ST ZIP   DATEFST      YRGOAL CONTACT
----- --------------- --------------- ---------- -- ----- --------- --------- ------------
11111 Stephen Daniels 123 Elm St      Seekonk    MA 02345 03-JUL-98       500 John Smith
12121 Jennifer Ames   24 Benefit St   Providence RI 02045 24-MAY-97       400 Susan Jones
22222 Carl Hersey     24 Benefit St   Providence RI 02045 03-JAN-98           Susan Jones
23456 Susan Ash       21 Main St      Fall River MA 02720 04-MAR-92       100 Amy Costa
33333 Nancy Taylor    26 Oak St       Fall River MA 02720 04-MAR-92        50 John Adams
34567 Robert Brooks   36 Pine St      Fall River MA 02720 04-APR-98        50 Amy Costa

6 rows selected.

SQL> DESC donor;
 Name                            Null?    Type
 ------------------------------- -------- ----
 IDNO                            NOT NULL VARCHAR2(5)
 NAME                                     VARCHAR2(15)
 STADR                                    VARCHAR2(15)
 CITY                                     VARCHAR2(10)
 STATE                                    VARCHAR2(2)
 ZIP                                      VARCHAR2(5)
 DATEFST                                  DATE
 YRGOAL                                   NUMBER(7,2)
 CONTACT                                  VARCHAR2(12)
```

SQL CODE: DONATION Table

```
SQL> SELECT * from donation;

IDNO  DRI CONTDATE    CONTAMT
----- --- --------- ---------
11111 100 07-JAN-99        25
12121 200 23-FEB-99        40
23456 100 03-MAR-99        20
33333 300 10-MAR-99        10

SQL> DESC donation;
 Name                            Null?    Type
 ------------------------------- -------- ----
 IDNO                                     VARCHAR2(5)
 DRIVENO                                  VARCHAR2(3)
 CONTDATE                                 DATE
 CONTAMT                                  NUMBER(6,2)
```

SQL CODE: DRIVE Table

Type here to search

9:33 AM
9/19/2019

Title: Sep 19-9:27 AM (1 of 31)

File Edit View History Bookmarks Tools Help

Multiple tables

www.pgrocer.net/Cis50/multiple.html

```
 ZIP                                      VARCHAR2(5)
 DATEFST                                  DATE
 YRGOAL                                   NUMBER(7,2)
 CONTACT                                  VARCHAR2(12)
```

SQL CODE: DONATION Table

```
SQL> SELECT * from donation;

IDNO  DRI CONTDATE    CONTAMT
----- --- --------- ---------
11111 100 07-JAN-99        25
12121 200 23-FEB-99        40
23456 100 03-MAR-99        20
33333 300 10-MAR-99        10

SQL> DESC donation;
 Name                            Null?    Type
 ------------------------------- -------- ----
 IDNO                                     VARCHAR2(5)
 DRIVENO                                  VARCHAR2(3)
 CONTDATE                                 DATE
 CONTAMT                                  NUMBER(6,2)
```

PK

SQL CODE: DRIVE Table

```
SQL> SELECT * from drive;

DRI DRIVENAME       DRIVECHAIR    LASTYEAR  THISYEAR
--- --------------- ------------ --------- ---------
100 Kids Shelter    Ann Smith        10000         0
200 Animal Home     Linda Grant       5000         0
300 Health Aid      David Ross        7000         0

SQL> DESC drive;
 Name                            Null?    Type
 ------------------------------- -------- ----
 DRIVENO                                  VARCHAR2(3)
 DRIVENAME                                VARCHAR2(15)
 DRIVECHAIR                               VARCHAR2(12)
 LASTYEAR                                 NUMBER(8,2)
 THISYEAR                                 NUMBER(8,2)

SQL>
```

PK

In a relational database, frequently you need to connect the data in two or more tables. The process to do this is called a JOIN. A simple join condition joins rows in one table to appropriate rows in another table based on a common value that exists in both tables. One of the basic ways of doing this follows the following format:

SELECT table1or2.column, table1or2.column
FROM table1, table2
WHERE table1.column = table2.column;

When you SELECT columns, you do not need to use the table name unless the column name appears in both tables, then to avoid confusion the table name must be used. In the FROM clause, the tables that are being used are listed. In the WHERE clause you make the connection between the tables. This is done by comparing the column that is common to both tables looking for a match. In the example below, I am joining the driveno in the donation table to the driveno in the drive table so I can get the name of the drive to display in my query results.

Type here to search

9:35 AM
9/19/2019

Title: Sep 19-9:35 AM (2 of 31)

File Edit View History Bookmarks Tools Help

Multiple tables

www.pgrocer.net/Cis50/multiple.html

```
SQL> SELECT * from drive;

DRI DRIVENAME       DRIVECHAIR    LASTYEAR  THISYEAR
--- --------------- ------------ --------- ---------
100 Kids Shelter    Ann Smith        10000         0
200 Animal Home     Linda Grant       5000         0
300 Health Aid      David Ross        7000         0


SQL> DESC drive;
 Name                            Null?    Type
 ------------------------------- -------- ----
 DRIVENO                                  VARCHAR2(3)
 DRIVENAME                                VARCHAR2(15)
 DRIVECHAIR                               VARCHAR2(12)
 LASTYEAR                                 NUMBER(8,2)
 THISYEAR                                 NUMBER(8,2)

SQL>
```

In a relational database, frequently you need to connect the data in two or more tables. The process to do this is called a JOIN. A simple join condition joins rows in one table to appropriate rows in another table based on a common value that exists in both tables. One of the basic ways of doing this follows the following format:

SELECT table1or2.column, table1or2.column
FROM table1, table2
WHERE table1.column = table2.column;

When you SELECT columns, you do not need to use the table name unless the column name appears in both tables, then to avoid confusion the table name must be used. In the FROM clause, the tables that are being used are listed. In the WHERE clause you make the connection between the tables. This is done by comparing the column that is common to both tables looking for a match. In the example below, I am joining the driveno in the donation table to the driveno in the drive table so I can get the name of the drive to display in my query results.

SQL CODE:

```
SQL> SELECT idno, donation.driveno, drivename
  2  FROM donation, drive
  3  WHERE donation.driveno = drive.driveno;

IDNO  DRI DRIVENAME
----- --- ---------------
11111 100 Kids Shelter
23456 100 Kids Shelter
12121 200 Animal Home
33333 300 Health Aid
```

If the where clause is omitted, then all possible connections are made (all rows in the first table are joined to all rows in the second table) and the results are meaningless as illustrated in the example below. This is called a Cartesian product and can be avoided by including a valid join condition.

SQL CODE:

```
SQL> SELECT idno, donation.driveno, drive.driveno, drivename
  2  FROM donation, drive;

IDNO  DRI DRI DRIVENAME
----- --- --- ---------------
```

Type here to search

9:40 AM
9/19/2019

File Edit View History Bookmarks Tools Help

Multiple tables

www.pgrocer.net/Cis50/multiple.html

SQL CODE:

```
SQL> SELECT idno, donation.driveno, drivename
  2  FROM donation, drive
  3  WHERE donation.driveno = drive.driveno;

IDNO  DRI DRIVENAME
----- --- ---------------
11111 100 Kids Shelter
23456 100 Kids Shelter
12121 200 Animal Home
33333 300 Health Aid
```

If the where clause is omitted, then all possible connections are made (all rows in the first table are joined to all rows in the second table) and the results are meaningless as illustrated in the example below. This is called a Cartesian product and can be avoided by including a valid join condition.

SQL CODE:

```
SQL> SELECT idno, donation.driveno, drive.driveno, drivename
  2  FROM donation, drive;

IDNO  DRI DRI DRIVENAME
----- --- --- ---------------
11111 100 100 Kids Shelter
12121 200 100 Kids Shelter
23456 100 100 Kids Shelter
33333 300 100 Kids Shelter
11111 100 200 Animal Home
12121 200 200 Animal Home
23456 100 200 Animal Home
33333 300 200 Animal Home
11111 100 300 Health Aid
12121 200 300 Health Aid
23456 100 300 Health Aid
33333 300 300 Health Aid

12 rows selected.
```

When joining tables, you can make up table aliases so that instead of having to refer to the entire table by name, you can used the table alias. The table aliases can be as short as one character. In the example below, I gave the donation table an alias of do and the drive table an alias of dr. I then used these aliases in the select statement and in the where statement that joined the tables. The rule for aliases is that if they are used in the FROM clause, they must be used throughout the SELECT.

SQL CODE:

```
SQL> SELECT idno, do.driveno, dr.driveno, drivename
  2  FROM donation do, drive dr
  3  WHERE do.driveno = dr.driveno;

IDNO  DRI DRI DRIVENAME
----- --- --- ---------------
11111 100 100 Kids Shelter
23456 100 100 Kids Shelter
12121 200 200 Animal Home
33333 300 300 Health Aid

SQL>
```

Type here to search

9:42 AM 9/19/2019

```
33333 300 200 Animal Home
11111 100 300 Health Aid
12121 200 300 Health Aid
23456 100 300 Health Aid
33333 300 300 Health Aid

12 rows selected.
```

When joining tables, you can make up table aliases so that instead of having to refer to the entire table by name, you can used the table alias. The table aliases can be as short as one character. In the example below, I gave the donation table an alias of do and the drive table an alias of dr. I then used these aliases in the select statement and in the where statement that joined the tables. The rule for aliases is that if they are used in the FROM clause, they must be used throughout the SELECT.

SQL CODE:

```
SQL> SELECT idno, do.driveno, dr.driveno, drivename
  2  FROM donation do, drive dr
  3  WHERE do.driveno = dr.driveno;

IDNO  DRI DRI DRIVENAME
----- --- --- ---------------
11111 100 100 Kids Shelter
23456 100 100 Kids Shelter
12121 200 200 Animal Home
33333 300 300 Health Aid

SQL>
```

There can be additional conditions included with the join condition that limit the range of the selection. In the example below, the WHERE clause joins the two tables and also limits the drives that are selected with the clause do.driveno > '100'.

SQL CODE:

```
SQL> SELECT idno, do.driveno, dr.driveno, drivename
  2  FROM donation do, drive dr
  3  WHERE do.driveno = dr.driveno AND do.driveno > '100';

IDNO  DRI DRI DRIVENAME
----- --- --- ---------------
12121 200 200 Animal Home
33333 300 300 Health Aid

SQL>
```

In the example below, I am joining three tables. Note that at this stage, all of the tables are in a one-to-one relationship through the idno which will join the donor table to the donation table and through the driveno which will join the donation table to the drive table. As soon as a donor makes a second contribution, the one-to-one relationship based on simply idno will no longer exist. In this example I have put table names in front of idno and driveno because these names exist in more than one table. The other column names stand alone because they occur only in one table, therefore confusion about which name is not a problem. In the from clause I have listed all three tables involved. In the where clause I have joined the donor table to the donation table and then joined the donation table to the drive table.

SQL CODE:

```
  1  SELECT donor.idno, name, donation.driveno, drivename, contamt
2 FROM donor, donation, drive
3* WHERE donor.idno = donation.idno and donation.driveno = drive.driveno
SQL> /
```

```
SQL>
```

In the example below, I am joining three tables. Note that at this stage, all of the tables are in a one-to-one relationship through the idno which will join the donor table to the donation table and through the driveno which will join the donation table to the drive table. As soon as a donor makes a second contribution, the one-to-one relationship based on simply idno will no longer exist. In this example I have put table names in front of idno and driveno because these names exist in more than one table. The other column names stand alone because they occur only in one table, therefore confusion about which name is not a problem. In the from clause I have listed all three tables involved. In the where clause I have joined the donor table to the donation table and then joined the donation table to the drive table.

SQL CODE:

```
  1  SELECT donor.idno, name, donation.driveno, drivename, contamt
2 FROM donor, donation, drive
3* WHERE donor.idno = donation.idno and donation.driveno = drive.driveno
SQL> /

IDNO  NAME            DRI DRIVENAME       CONTAMT
----- --------------- --- --------------- ---------
11111 Stephen Daniels 100 Kids Shelter           25
23456 Susan Ash       100 Kids Shelter           20
12121 Jennifer Ames   200 Animal Home            40
33333 Nancy Taylor    300 Health Aid             10
```

Donor   donation   Drive
Idno ⟷ Idno
Driveno ⟷ Driveno

JOIN Explanation:

The joins that we are going to look at are equijoins, non-equijoins, outer joins and self joins.

Equijoin

An equijoin is the kind that has been illustrated so far in this handout. Other names for equijoins are simple joins or inner joins. An equijoin means that values in the tables being compared must be equal.

Non-equijoin

A non-equijoin means that there is not a direct join between two tables. The relationship might be that you relate a number on one table to a range on another. For example one table might have a student's grade and the other might have a range of 1 thru 6 for elementary, 7 thru 8 for junior high etc. You could then compare the student's grade to the low and high for each grade in the range and determine the group the student is in.

```
SELECT grade.name, grade.gradelevel, schools.groupname
FROM grade, schools
WHERE grade.gradelevel BETWEEN schools.lowgrade AND schools.highgrade;
```

Outer join

An outer join can be used to display records that do not satisfy the join condition. To do this, we put the join operator which is a plus sign in parenthesis in the where clause on the side of the join that is missing the information. The result is the creation of null rows on the side that is missing the information which are joined with the side that contains information you need to see. For this example, I added a record to the drive table that contained a drive that did not exist in the donation table because no one had given to that drive.

SQL CODE:

File Edit View History Bookmarks Tools Help

Multiple tables

www.pgrocer.net/Cis50/multiple.html

JOIN Explanation:

The joins that we are going to look at are equijoins, non-equijoins, outer joins and self joins.

Equijoin

An equijoin is the kind that has been illustrated so far in this handout. Other names for equijoins are simple joins or inner joins. An equijoin means that values in the tables being compared must be equal.

Non-equijoin

A non-equijoin means that there is not a direct join between two tables. The relationship might be that you relate a number on one table to a range on another. For example one table might have a student's grade and the other might have a range of 1 thru 6 for elementary, 7 thru 8 for junior high etc. You could then compare the student's grade to the low and high for each grade in the range and determine the group the student is in.

SELECT grade.name, grade.gradelevel, schools.groupname
FROM grade, schools
WHERE grade.gradelevel BETWEEN schools.lowgrade AND schools.highgrade;

11 9 12 high school

Outer join

An outer join can be used to display records that do not satisfy the join condition. To do this, we put the join operator which is a plus sign in parenthesis in the where clause on the side of the join that is missing the information. The result is the creation of null rows on the side that is missing the information which are joined with the side that contains information you need to see. For this example, I added a record to the drive table that contained a drive that did not exist in the donation table because no one had given to that drive.

SQL CODE:

```
SQL> select * from drive;

DRI DRIVENAME       DRIVECHAIR    LASTYEAR  THISYEAR
--- --------------- ------------ --------- ---------
100 Kids Shelter    Ann Smith        10000         0
200 Animal Home     Linda Grant       5000         0
300 Health Aid      David Ross        7000         0

SQL> INSERT INTO drive
  2  values('400', 'Half Way', 'Robert Doe', 0,0);

1 row created.

SQL> SELECT * from drive;

DRI DRIVENAME       DRIVECHAIR    LASTYEAR  THISYEAR
--- --------------- ------------ --------- ---------
100 Kids Shelter    Ann Smith        10000         0
200 Animal Home     Linda Grant       5000         0
300 Health Aid      David Ross        7000         0
400 Half Way        Robert Doe           0         0

SQL>
```

Type here to search

9:48 AM
9/19/2019

File Edit View History Bookmarks Tools Help

Multiple tables

www.pgrocer.net/Cis50/multiple.html

Outer join

An outer join can be used to display records that do not satisfy the join condition. To do this, we put the join operator which is a plus sign in parenthesis in the where clause on the side of the join that is missing the information. The result is the creation of null rows on the side that is missing the information which are joined with the side that contains information you need to see. For this example, I added a record to the drive table that contained a drive that did not exist in the donation table because no one had given to that drive.

SQL CODE:

```
SQL> select * from drive;

DRI DRIVENAME       DRIVECHAIR    LASTYEAR  THISYEAR
--- --------------- ------------ --------- ---------
100 Kids Shelter    Ann Smith        10000         0
200 Animal Home     Linda Grant       5000         0
300 Health Aid      David Ross        7000         0

SQL> INSERT INTO drive
  2  values('400', 'Half Way', 'Robert Doe', 0,0);

1 row created.

SQL> SELECT * from drive;

DRI DRIVENAME       DRIVECHAIR    LASTYEAR  THISYEAR
--- --------------- ------------ --------- ---------
100 Kids Shelter    Ann Smith        10000         0
200 Animal Home     Linda Grant       5000         0
300 Health Aid      David Ross        7000         0
400 Half Way        Robert Doe           0         0

SQL>
```

The code below illustrates the outer join. There is now a drive 400, but there have been no contributions to the drive. Therefore the donation side is missing the data while the drive side has the data. Therefore in setting up the test, the (+) is put on the donation side. The second example below eliminates the (+) on the donation side and therefore the drive 400 does not display.

SQL CODE:

```
SQL> SELECT do.driveno, do.contamt, dr.driveno, dr.drivename
  2  FROM donation do, drive dr
  3  WHERE do.driveno(+) = dr.driveno;

DRI   CONTAMT DRI DRIVENAME
--- --------- --- ---------------
100        25 100 Kids Shelter
100        20 100 Kids Shelter
200        40 200 Animal Home
300        10 300 Health Aid
              400 Half Way

SQL> SELECT do.driveno, do.contamt, dr.driveno, dr.drivename
  2  FROM donation do, drive dr
  3  WHERE do.driveno = dr.driveno;
```

Type here to search

9:49 AM
9/19/2019

```
DRI DRIVENAME       DRIVECHAIR    LASTYEAR  THISYEAR
--- --------------- ------------ --------- ---------
100 Kids Shelter    Ann Smith        10000         0
200 Animal Home     Linda Grant       5000         0
300 Health Aid      David Ross        7000         0
400 Half Way        Robert Doe           0         0

SQL>
```

The code below illustrates the outer join. There is now a drive 400, but there have been no contributions to the drive. Therefore the donation side is missing the data while the drive side has the data. Therefore in setting up the test, the (+) is put on the donation side. The second example below eliminates the (+) on the donation side and therefore the drive 400 does not display.

SQL CODE:

```
SQL> SELECT do.driveno, do.contamt, dr.driveno, dr.drivename
  2  FROM donation do, drive dr
  3  WHERE do.driveno(+) = dr.driveno;

DRI   CONTAMT DRI DRIVENAME
--- --------- --- ---------------
100        25 100 Kids Shelter
100        20 100 Kids Shelter
200        40 200 Animal Home
300        10 300 Health Aid
              400 Half Way

SQL> SELECT do.driveno, do.contamt, dr.driveno, dr.drivename
  2  FROM donation do, drive dr
  3  WHERE do.driveno = dr.driveno;

DRI   CONTAMT DRI DRIVENAME
--- --------- --- ---------------
100        25 100 Kids Shelter
100        20 100 Kids Shelter
200        40 200 Animal Home
300        10 300 Health Aid

SQL>
```

Self join

A self join is used to join a table to itself. For example, let's say you have an inventory file that contains jackets and skirts that can be sold separately or together as a suit. You will carry the coordinate in a field called pairset.

SQL CODE:

```
SQL> SELECT* FROM inventry;

ITE ITEMNAME   PAI
--- ---------- ---
111 jacket     333
222 jacket     555
333 skirt      111
444 blouse
```

File Edit View History Bookmarks Tools Help

Multiple tables

www.pgrocer.net/Cis50/multiple.html

```
100        25 100 Kids Shelter
100        20 100 Kids Shelter
200        40 200 Animal Home
300        10 300 Health Aid
              400 Half Way

SQL> SELECT do.driveno, do.contamt, dr.driveno, dr.drivename
  2  FROM donation do, drive dr
  3  WHERE do.driveno = dr.driveno;

DRI   CONTAMT DRI DRIVENAME
--- --------- --- ---------------
100        25 100 Kids Shelter
100        20 100 Kids Shelter
200        40 200 Animal Home
300        10 300 Health Aid

SQL>
```

Self join

A self join is used to join a table to itself. For example, let's say you have an inventory file that contains jackets and skirts that can be sold separately or together as a suit. You will carry the coordinate in a field called pairset.

SQL CODE:

```
SQL> SELECT* FROM inventry;

ITE ITEMNAME   PAI
--- ---------- ---
111 jacket     333
222 jacket     555
333 skirt      111
444 blouse
555 shirt      222

SQL> DESC inventry
 Name                            Null?    Type
 ------------------------------- -------- ----
 ITEMNO                                   VARCHAR2(3)
 ITEMNAME                                 VARCHAR2(10)
 PAIRSET                                  VARCHAR2(3)

SQL> list
  1  SELECT fst.itemno, fst.itemname, snd.itemname
  2  FROM inventry fst, inventry snd
  3* WHERE fst.itemno = snd.pairset
SQL> /

ITE ITEMNAME   ITEMNAME
--- ---------- ----------
111 jacket     skirt
222 jacket     shirt
333 skirt      jacket
555 shirt      jacket
```

Type here to search

9:53 AM
9/19/2019

Title: Sep 19-9:53 AM (10 of 31)

File Edit View History Bookmarks Tools Help

Relational database

www.pgrocer.net/Cis50/relorder.html

```
1111 Good Night Moon          24        30        40         8     12.99 BK BY X100
1212 Heidi                    12        25        25        10     14.99 BK CH X112
1234 Adven Reddy Fox           5         0        10         9     14.75 BK CH X100
2121 Teddy Bear                5        20        40        15     19.95 TY CH X115
2222 Building Blocks           4         0        15        23     27.98 TY CH Z200
2345 Doll House                2         5        10        45     55.98 TY CH Z212
3333 Basketball               24        25        50        14     17.99 SP BK Y200
3456 Net/Hoop                 10         0        25        25     27.95 SP BK Y200

8 rows selected.

SQL> desc invcust;
 Name                            Null?    Type
 ------------------------------- -------- ----
 CUSTID                                   VARCHAR2(5)
 CUSTNAME                                 VARCHAR2(20)
 STADR                                    VARCHAR2(5)
 APT                                      VARCHAR2(5)
 CITY                                     VARCHAR2(15)
 STATE                                    CHAR(2)
 ZIP                                      VARCHAR2(5)
 PASTDUE                                  NUMBER(6,2)
 CURRDUE                                  NUMBER(6,2)
 CRLIMIT                                  NUMBER(6,2)
 DATEFST                                  DATE
 SLSREP                                   VARCHAR2(4)

SQL> select * from invcust;

CUSTI CUSTNAME             STADR           APT   CITY            ST ZIP     PASTDUE   CURRDUE
----- -------------------- --------------- ----- --------------- -- ----- --------- ---------
  CRLIMIT DATEFST   SLSR
--------- --------- ----
11111 Susan Ash            123 Elm St            Fall River      MA 02720         0         0
     1000 05-NOV-97 1111

12121 Richard Davis        24 West St            Fall River      MA 02720         0       100
      500 10-DEC-98 1212

12345 Linda Anderson       45 Main St      A#3   Seekonk         MA 02771       100         0
      500 14-OCT-98 1111

SQL> desc orderz
 Name                            Null?    Type
 ------------------------------- -------- ----
 ORDNO                                    VARCHAR2(6)
 CUSTID                                   VARCHAR2(5)
 ORDATE                                   DATE

SQL> select * from orderz;

ORDNO  CUSTI ORDATE
------ ----- ---------
000001 11111 10-JUN-99
000002 12121 10-JUN-99
000003 12345 10-JUN-99

SQL> desc ordline
 Name                            Null?    Type
 ------------------------------- -------- ----
 ORDNO                                    VARCHAR2(6)
```

Inven

Itemno

Invcust

custid

Type here to search

9:55 AM
9/19/2019

File Edit View History Bookmarks Tools Help

Relational database

www.pgrocer.net/Cis50/relorder.html

```
 DATEFST                                  DATE
 SLSREP                                   VARCHAR2(4)

SQL> select * from invcust;

CUSTI CUSTNAME            STADR          APT   CITY            ST ZIP      PASTDUE   CURRDUE
----- ------------------- -------------- ----- --------------- -- ----- --------- ---------
  CRLIMIT DATEFST   SLSR
--------- --------- ----
11111 Susan Ash           123 Elm St           Fall River      MA 02720         0         0
     1000 05-NOV-97 1111

12121 Richard Davis       24 West St           Fall River      MA 02720         0       100
      500 10-DEC-98 1212

12345 Linda Anderson      45 Main St     A#3   Seekonk         MA 02771       100         0
      500 14-OCT-98 1111

SQL> desc orderz
 Name                            Null?    Type
 ------------------------------- -------- ----
 ORDNO                                    VARCHAR2(6)
 CUSTID                                   VARCHAR2(5)
 ORDATE                                   DATE

SQL> select * from orderz;

ORDNO  CUSTI ORDATE
------ ----- ---------
000001 11111 10-JUN-99
000002 12121 10-JUN-99
000003 12345 10-JUN-99

SQL> desc ordline
 Name                            Null?    Type
 ------------------------------- -------- ----
 ORDNO                                    VARCHAR2(6)
 ITEMNO                                   VARCHAR2(4)
 NUMORD                                   NUMBER(3)

SQL> select * from ordline;

ORDNO  ITEM    NUMORD
------ ---- ---------
000001 1212         1
000001 2121         1
000001 2345         1
000002 1111         3
000002 3333         1
000003 3333         2
000003 3456         1

7 rows selected.

SQL> desc salsrep;
 Name                            Null?    Type
 ------------------------------- -------- ----
 SALSREP                                  VARCHAR2(4)
 SALSNAME                                 VARCHAR2(20)
 COMMRATE                                 NUMBER(3,2)
```

Inven
Itemno

Invcust
Custid

Ordline
Ordno
Itemno
Numord

Orderz
Ordno
Custid
Ordate

Type here to search

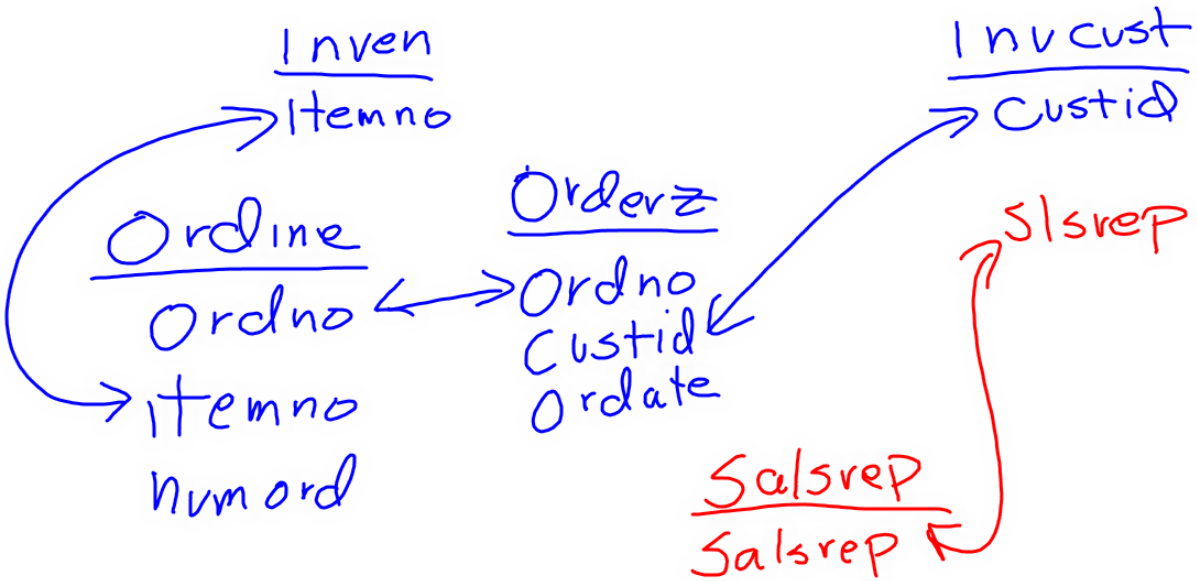
10:01 AM 9/19/2019

File Edit View History Bookmarks Tools Help

Relational database

www.pgrocer.net/Cis50/relorder.html

```
SQL> select * from ordline;

ORDNO  ITEM     NUMORD
------ ---- ---------
000001 1212         1
000001 2121         1
000001 2345         1
000002 1111         3
000002 3333         1
000003 3333         2
000003 3456         1

7 rows selected.

SQL> desc salsrep;
 Name                            Null?    Type
 ------------------------------- -------- ----
 SALSREP                                  VARCHAR2(4)
 SALSNAME                                 VARCHAR2(20)
 COMMRATE                                 NUMBER(3,2)

SQL> select * from salsrep;

SALS SALSNAME             COMMRATE
---- -------------------- ---------
1111 John Smith                 .08
1212 Joanne London              .09

SQL> desc department
 Name                            Null?    Type
 ------------------------------- -------- ----
 DEPT                                     CHAR(2)
 DEPTNAME                                 VARCHAR2(15)
 MANAGER                                  VARCHAR2(15)

SQL> select * from department;

DE DEPTNAME        MANAGER
-- --------------- ---------------
BK Books           Jennifer Moore
SP Sports          Stephen Willis
TY Toys            Carl Baker
```



Doing queries using information from multiple tables:

In this example, I am taking the I want to print the item number, item name, and dept from the inven table and then the deptname and manager from the department table. I am relating the two tables on the thing they have in common which is dept.

SQL CODE:

```
SQL> SELECT itemno, itemname, inven.dept, deptname, manager
  2  FROM inven, department
  3  WHERE inven.dept = department.dept;

ITEM ITEMNAME           DE DEPTNAME         MANAGER
```

Type here to search

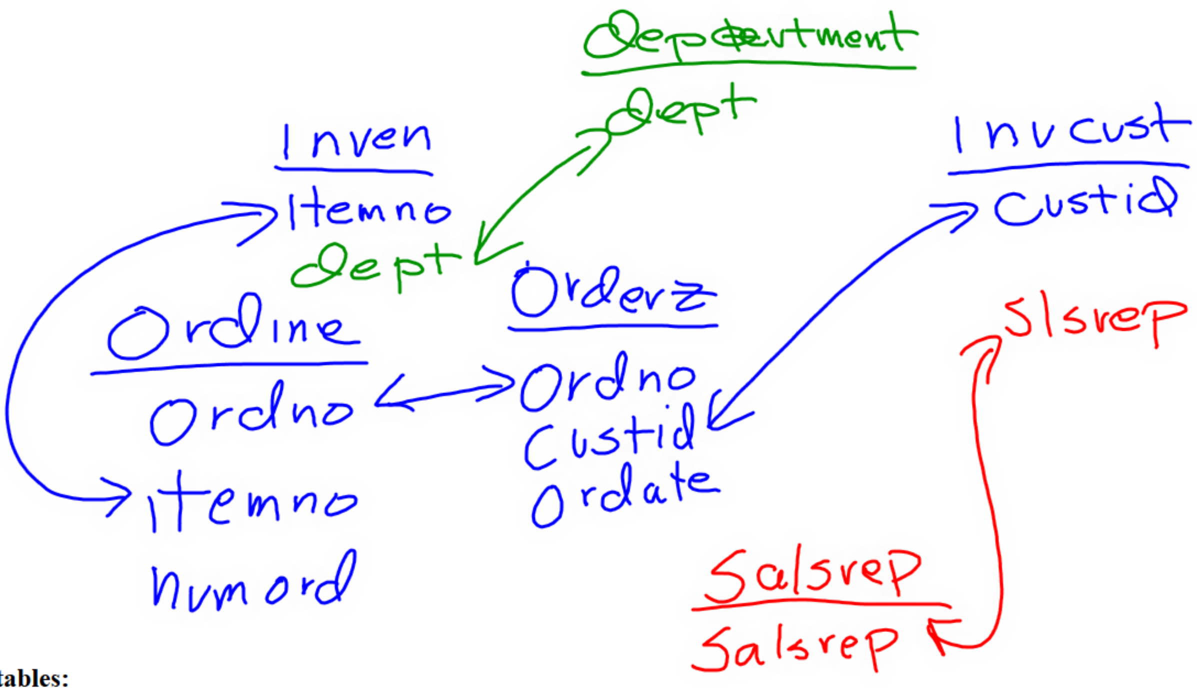
10:04 AM
9/19/2019

File Edit View History Bookmarks Tools Help

Relational database

www.pgrocer.net/Cis50/relorder.html

```
000001 1212         1
000001 2121         1
000001 2345         1
000002 1111         3
000002 3333         1
000003 3333         2
000003 3456         1

7 rows selected.

SQL> desc salsrep;
 Name                            Null?    Type
 ------------------------------- -------- ----
 SALSREP                                  VARCHAR2(4)
 SALSNAME                                 VARCHAR2(20)
 COMMRATE                                 NUMBER(3,2)

SQL> select * from salsrep;

SALS SALSNAME             COMMRATE
---- -------------------- ---------
1111 John Smith                 .08
1212 Joanne London              .09

SQL> desc department
 Name                            Null?    Type
 ------------------------------- -------- ----
 DEPT                                     CHAR(2)
 DEPTNAME                                 VARCHAR2(15)
 MANAGER                                  VARCHAR2(15)

SQL> select * from department;

DE DEPTNAME        MANAGER
-- --------------- ---------------
BK Books           Jennifer Moore
SP Sports          Stephen Willis
TY Toys            Carl Baker
```



Doing queries using information from multiple tables:

In this example, I am taking the I want to print the item number, item name, and dept from the inven table and then the deptname and manager from the department table. I am relating the two tables on the thing they have in common which is dept.

SQL CODE:

```
SQL> SELECT itemno, itemname, inven.dept, deptname, manager
  2  FROM inven, department
  3  WHERE inven.dept = department.dept;

ITEM ITEMNAME        DE DEPTNAME        MANAGER
---- --------------- -- --------------- ---------------
1111 Good Night Moon BK Books           Jennifer Moore
1212 Heidi           BK Books           Jennifer Moore
1234 Adven Reddy Fox BK Books           Jennifer Moore
3333 Basketball      SP Sports          Stephen Willis
```

Type here to search

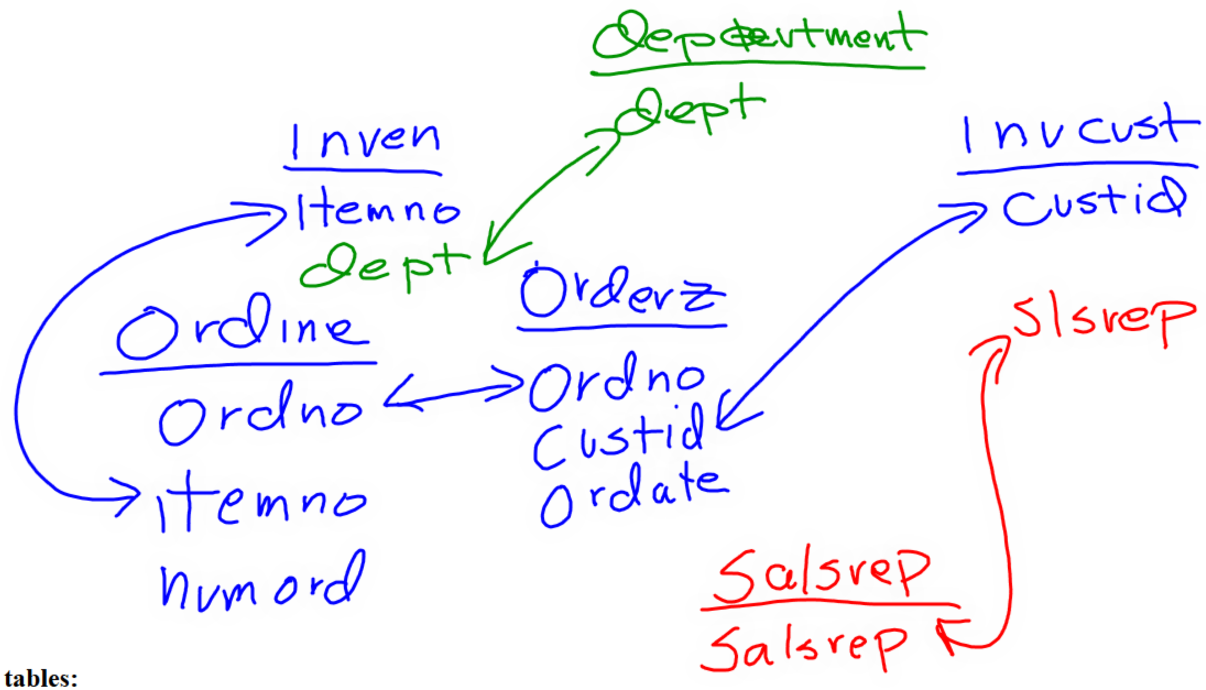
10:06 AM
9/19/2019

File Edit View History Bookmarks Tools Help

Relational database

www.pgrocer.net/Cis50/relorder.html

```
000001 1212         1
000001 2121         1
000001 2345         1
000002 1111         3
000002 3333         1
000003 3333         2
000003 3456         1

7 rows selected.

SQL> desc salsrep;
 Name                            Null?    Type
 ------------------------------- -------- ----
 SALSREP                                  VARCHAR2(4)
 SALSNAME                                 VARCHAR2(20)
 COMMRATE                                 NUMBER(3,2)

SQL> select * from salsrep;

SALS SALSNAME             COMMRATE
---- -------------------- ---------
1111 John Smith                 .08
1212 Joanne London              .09

SQL> desc department
 Name                            Null?    Type
 ------------------------------- -------- ----
 DEPT                                     CHAR(2)
 DEPTNAME                                 VARCHAR2(15)
 MANAGER                                  VARCHAR2(15)

SQL> select * from department;

DE DEPTNAME        MANAGER
-- --------------- ---------------
BK Books           Jennifer Moore
SP Sports          Stephen Willis
TY Toys            Carl Baker
```



Doing queries using information from multiple tables:

In this example, I am taking the I want to print the item number, item name, and dept from the inven table and then the deptname and manager from the department table. I am relating the two tables on the thing they have in common which is dept.

SQL CODE:

```
SQL> SELECT itemno, itemname, inven.dept, deptname, manager
  2  FROM inven, department
  3  WHERE inven.dept = department.dept;

ITEM ITEMNAME        DE DEPTNAME        MANAGER
---- --------------- -- --------------- ---------------
1111 Good Night Moon BK Books           Jennifer Moore
1212 Heidi           BK Books           Jennifer Moore
1234 Adven Reddy Fox BK Books           Jennifer Moore
3333 Basketball      SP Sports          Stephen Willis
```

Type here to search

10:06 AM
9/19/2019

```
TY Toys             Carl Baker
```

Doing queries using information from multiple tables:

In this example, I am taking the I want to print the item number, item name, and dept from the inven table and then the deptname and manager from the department table. I am relating the two tables on the thing they have in common which is dept.

SQL CODE:

```
SQL> SELECT itemno, itemname, inven.dept, deptname, manager
  2  FROM inven, department
  3  WHERE inven.dept = department.dept;

ITEM ITEMNAME        DE DEPTNAME        MANAGER
---- --------------- -- --------------- ---------------
1111 Good Night Moon BK Books           Jennifer Moore
1212 Heidi           BK Books           Jennifer Moore
1234 Adven Reddy Fox BK Books           Jennifer Moore
3333 Basketball      SP Sports          Stephen Willis
3456 Net/Hoop        SP Sports          Stephen Willis
2121 Teddy Bear      TY Toys            Carl Baker
2222 Building Blocks TY Toys            Carl Baker
2345 Doll House      TY Toys            Carl Baker

8 rows selected.
```

In this example, I am taking the order header information and adding the customer name from the invcust file.

SQL CODE:

```
SQL> SELECT orderz.ordno, orderz.custid, custname, ordate
  2  FROM orderz, invcust
  3  WHERE orderz.custid = invcust.custid;

ORDNO  CUSTI CUSTNAME             ORDATE
------ ----- -------------------- ---------
000001 11111 Susan Ash            10-JUN-99
000002 12121 Richard Davis        10-JUN-99
000003 12345 Linda Anderson       10-JUN-99
```

This example is showing the order information from the order header and the order detail or line item file.

SQL CODE:

```
1  SELECT orderz.ordno, custid, ordate, itemno, numord
  2  FROM orderz, ordline
  3* WHERE orderz.ordno = ordline.ordno
SQL> /

ORDNO  CUSTI ORDATE    ITEM    NUMORD
```

Department
dept

Inven
Itemno
dept

Invcust
Custid

Orderz
Ordno
Custid
Ordate

Ordline
Ordno
Itemno
Numord

Slsrep

Salsrep
Salsrep

File Edit View History Bookmarks Tools Help

Relational database

www.pgrocer.net/Cis50/relorder.html

```
SQL> SELECT orderz.ordno, orderz.custid, custname, ordate
  2  FROM orderz, invcust
  3  WHERE orderz.custid = invcust.custid;

ORDNO  CUSTI CUSTNAME             ORDATE
------ ----- -------------------- ---------
000001 11111 Susan Ash            10-JUN-99
000002 12121 Richard Davis        10-JUN-99
000003 12345 Linda Anderson       10-JUN-99
```

This example is showing the order information from the order header and the order detail or line item file.

SQL CODE:

```
1  SELECT orderz.ordno, custid, ordate, itemno, numord
  2  FROM orderz, ordline
  3* WHERE orderz.ordno = ordline.ordno
SQL> /

ORDNO  CUSTI ORDATE    ITEM    NUMORD
------ ----- --------- ---- ---------
000001 11111 10-JUN-99 1212         1
000001 11111 10-JUN-99 2121         1
000001 11111 10-JUN-99 2345         1
000002 12121 10-JUN-99 1111         3
000002 12121 10-JUN-99 3333         1
000003 12345 10-JUN-99 3333         2
000003 12345 10-JUN-99 3456         1

7 rows selected.
```

In this example, I am using three tables to get the information I need. To do this I put two conditions in the where, one to relate the orderz table to the ordline table based on ordno and the other to relate the ordline table to the inven table based on itemno. In the second example, I gave each table name an alias which allowed less code.

SQL CODE:

```
1  SELECT orderz.ordno, ordline.itemno, itemname, numord
  2  FROM orderz, ordline, inven
  3* WHERE orderz.ordno = ordline.ordno and ordline.itemno = inven.itemno
SQL> /

ORDNO  ITEM ITEMNAME        NUMORD
------ ---- --------------- ---------
000001 1212 Heidi                   1
000001 2121 Teddy Bear              1
000001 2345 Doll House              1
000002 1111 Good Night Moon         3
000002 3333 Basketball              1
000003 3333 Basketball              2
000003 3456 Net/Hoop                1

7 rows selected.
```

Department
Dept

Inven
Itemno
Dept

Invcust
Custid

Ordline
Ordno
itemno
Numord

Orderz
Ordno
Custid
Ordate
Salsrep

Slsrep
Salsrep

Type here to search

10:10 AM 9/19/2019

File Edit View History Bookmarks Tools Help

Relational database

www.pgrocer.net/Cis50/relorder.html

SQL CODE:

```
1  SELECT orderz.ordno, ordline.itemno, itemname, numord
  2  FROM orderz, ordline, inven
  3* WHERE orderz.ordno = ordline.ordno and ordline.itemno = inven.itemno
SQL> /

ORDNO  ITEM ITEMNAME              NUMORD
------ ---- --------------- ---------
000001 1212 Heidi                      1
000001 2121 Teddy Bear                 1
000001 2345 Doll House                 1
000002 1111 Good Night Moon            3
000002 3333 Basketball                 1
000003 3333 Basketball                 2
000003 3456 Net/Hoop                   1

7 rows selected.

SQL> SELECT h.ordno, l.itemno, itemname, numord
  2  FROM orderz h, ordline l, inven i
  3  WHERE h.ordno = l.ordno and l.itemno = i.itemno;

ORDNO  ITEM ITEMNAME              NUMORD
------ ---- --------------- ---------
000001 1212 Heidi                      1
000001 2121 Teddy Bear                 1
000001 2345 Doll House                 1
000002 1111 Good Night Moon            3
000002 3333 Basketball                 1
000003 3333 Basketball                 2
000003 3456 Net/Hoop                   1

7 rows selected.
```

Department

Dept

Inven

Itemno

dept

Invcust

Custid

Orderz

Ordno

Custid

Ordate

Slsrep

Ordline

Ordno

itemno

numord

Salsrep

This example uses four tables to get information. The information is in the order header, order line item as well as inventory and customer. Note that I have three conditions in the where to relate the tables.

Salsrep

SQL CODE:

```
SQL> SELECT orderz.ordno, orderz.custid, custname, ordate,      ordline.itemno, itemname, numord
  2  FROM orderz, invcust, inven, ordline
3 WHERE orderz.ordno = ordline.ordno and orderz.custid=invcust.custid and
4 ordline.itemno=inven.itemno;

ORDNO  CUSTI CUSTNAME             ORDATE    ITEM ITEMNAME            NUMORD
------ ----- -------------------- --------- ---- --------------- ---------
000001 11111 Susan Ash            10-JUN-99 1212 Heidi                   1
000001 11111 Susan Ash            10-JUN-99 2121 Teddy Bear              1
000001 11111 Susan Ash            10-JUN-99 2345 Doll House              1
000002 12121 Richard Davis        10-JUN-99 1111 Good Night Moon         3
000002 12121 Richard Davis        10-JUN-99 3333 Basketball              1
000003 12345 Linda Anderson       10-JUN-99 3333 Basketball              2
000003 12345 Linda Anderson       10-JUN-99 3456 Net/Hoop                1

7 rows selected.
```

Type here to search

10:11 AM 9/19/2019

File Edit View History Bookmarks Tools Help

Relational database

www.pgrocer.net/Cis50/relorder.html

```
7 rows selected.
```

This example uses four tables to get information. The information is in the order header, order line item as well as inventory and customer. Note that I have three conditions in the where to relate the tables.

Department

SQL CODE:

```
SQL> SELECT orderz.ordno, orderz.custid, custname, ordate,      ordline.itemno, itemname, numord
  2  FROM orderz, invcust, inven, ordline
3 WHERE orderz.ordno = ordline.ordno and orderz.custid=invcust.custid and
4 ordline.itemno=inven.itemno;

ORDNO  CUSTI CUSTNAME             ORDATE    ITEM ITEMNAME          NUMORD
------ ----- -------------------- --------- ---- --------------- ---------
000001 11111 Susan Ash            10-JUN-99 1212 Heidi                     1
000001 11111 Susan Ash            10-JUN-99 2121 Teddy Bear                1
000001 11111 Susan Ash            10-JUN-99 2345 Doll House                1
000002 12121 Richard Davis        10-JUN-99 1111 Good Night Moon           3
000002 12121 Richard Davis        10-JUN-99 3333 Basketball                1
000003 12345 Linda Anderson       10-JUN-99 3333 Basketball                2
000003 12345 Linda Anderson       10-JUN-99 3456 Net/Hoop                  1

7 rows selected.
```

Dept

Inven

Itemno

dept

Invcust

Custid

Ordline

Ordno

itemno

Numord

Orderz

Ordno

Custid

Ordate

Slsrep

Salsrep

Salsrep

This example also uses four tables. Note that there is also a claculation that is being done.

SQL CODE:

```
SQL> SELECT h.ordno, h.custid, custname, l.itemno, itemname, price, numord, price * numord amtdue
  2  FROM orderz h, ordline l, inven i, invcust c
  3  WHERE h.ordno = l.ordno and l.itemno = i.itemno and h.custid = c.custid;

ORDNO  CUSTI CUSTNAME             ITEM ITEMNAME            PRICE    NUMORD    AMTDUE
------ ----- -------------------- ---- --------------- --------- --------- ---------
000002 12121 Richard Davis        1111 Good Night Moon     12.99         3     38.97
000001 11111 Susan Ash            1212 Heidi               14.99         1     14.99
000001 11111 Susan Ash            2121 Teddy Bear          19.95         1     19.95
000001 11111 Susan Ash            2345 Doll House          55.98         1     55.98
000002 12121 Richard Davis        3333 Basketball          17.99         1     17.99
000003 12345 Linda Anderson       3333 Basketball          17.99         2     35.98
000003 12345 Linda Anderson       3456 Net/Hoop            27.95         1     27.95

7 rows selected.
```

Note that in the example above, the order is not on order number which you would expect. As I have found, the order is really only under your control if you use the order by clause as I have in the example below:

SQL CODE:

```
1  SELECT h.ordno, h.custid, custname, l.itemno, itemname, price, numord, price * numord amtdue
  2  FROM orderz h, ordline l, inven i, invcust c
  3  WHERE h.ordno = l.ordno and l.itemno = i.itemno and h.custid = c.custid
  4* ORDER BY h.ordno
```

Type here to search

10:12 AM 9/19/2019

File Edit View History Bookmarks Tools Help

Relational database

www.pgrocer.net/Cis50/relorder.html

```
------ ----- -------------------- --------- ---- --------------- ---------
000001 11111 Susan Ash            10-JUN-99 1212 Heidi                   1
000001 11111 Susan Ash            10-JUN-99 2121 Teddy Bear              1
000001 11111 Susan Ash            10-JUN-99 2345 Doll House              1
000002 12121 Richard Davis        10-JUN-99 1111 Good Night Moon         3
000002 12121 Richard Davis        10-JUN-99 3333 Basketball              1
000003 12345 Linda Anderson       10-JUN-99 3333 Basketball              2
000003 12345 Linda Anderson       10-JUN-99 3456 Net/Hoop                1

7 rows selected.
```

This example also uses four tables. Note that there is also a claculation that is being done.

SQL CODE:

```
SQL> SELECT h.ordno, h.custid, custname, l.itemno, itemname, price, numord, price * numord amtdue
  2  FROM orderz h, ordline l, inven i, invcust c
  3  WHERE h.ordno = l.ordno and l.itemno = i.itemno and h.custid = c.custid;

ORDNO  CUSTI CUSTNAME             ITEM ITEMNAME            PRICE    NUMORD    AMTDUE
------ ----- -------------------- ---- --------------- --------- --------- ---------
000002 12121 Richard Davis        1111 Good Night Moon     12.99         3     38.97
000001 11111 Susan Ash            1212 Heidi               14.99         1     14.99
000001 11111 Susan Ash            2121 Teddy Bear          19.95         1     19.95
000001 11111 Susan Ash            2345 Doll House          55.98         1     55.98
000002 12121 Richard Davis        3333 Basketball          17.99         1     17.99
000003 12345 Linda Anderson       3333 Basketball          17.99         2     35.98
000003 12345 Linda Anderson       3456 Net/Hoop            27.95         1     27.95

7 rows selected.
```

Note that in the example above, the order is not on order number which you would expect. As I have found, the order is really only under your control if you use the order by clause as I have in the example below:

SQL CODE:

```
1  SELECT h.ordno, h.custid, custname, l.itemno, itemname, price, numord, price * numord amtdue
  2  FROM orderz h, ordline l, inven i, invcust c
  3  WHERE h.ordno = l.ordno and l.itemno = i.itemno and h.custid = c.custid
  4* ORDER BY h.ordno
SQL> /

ORDNO  CUSTI CUSTNAME             ITEM ITEMNAME            PRICE    NUMORD    AMTDUE
------ ----- -------------------- ---- --------------- --------- --------- ---------
000001 11111 Susan Ash            1212 Heidi               14.99         1     14.99
000001 11111 Susan Ash            2121 Teddy Bear          19.95         1     19.95
000001 11111 Susan Ash            2345 Doll House          55.98         1     55.98
000002 12121 Richard Davis        1111 Good Night Moon     12.99         3     38.97
000002 12121 Richard Davis        3333 Basketball          17.99         1     17.99
000003 12345 Linda Anderson       3333 Basketball          17.99         2     35.98
000003 12345 Linda Anderson       3456 Net/Hoop            27.95         1     27.95

7 rows selected.
```

Department
Dept

Inven
Itemno
dept

Invcust
Custid

Orderz
Ordno
Custid
Ordate

Ordline
Ordno
Itemno
Numord

Slsrep

Salsrep
Salsrep

Type here to search

10:15 AM
9/19/2019

File Edit View History Bookmarks Tools Help

JOIN in Oracle:

www.pgrocer.net/Cis50/oraclejoin.html

```
20 RESEARCH DALLAS
30 SALES CHICAGO
40 OPERATIONS BOSTON
```

Using the join instead of the while.

Natural Join:

Now I am going to code the natural join. Note that I list the fields that I want in the query but I do associate the table name with the field they have in common, deptno. Rather than use a where clause, the natural join clause is used in the from.

```
SQL> select empno, ename, deptno, dname
2 from emp natural join dept;

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7369 SMITH 20 RESEARCH
7499 ALLEN 30 SALES
7521 WARD 30 SALES
7566 JONES 20 RESEARCH
7654 MARTIN 30 SALES
7698 BLAKE 30 SALES
7782 CLARK 10 ACCOUNTING
7788 SCOTT 20 RESEARCH
7839 KING 10 ACCOUNTING
7844 TURNER 30 SALES
7876 ADAMS 20 RESEARCH

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7900 JAMES 30 SALES
7902 FORD 20 RESEARCH
7934 MILLER 10 ACCOUNTING

14 rows selected.
```

When I tried it with the table name.field name, I got an error saying that I cannot have qualifiers on a column being used in a natural join. As we have note before, the deptno field/column in the emp table will be equal to the deptno field/column in the dept table by defination when they are linked so there is no need for the qualifier.

```
SQL> select empno, ename, emp.deptno, dept.deptno, dname
2 from emp natural join dept;
select empno, ename, emp.deptno, dept.deptno, dname
*
ERROR at line 1:
ORA-25155: column used in NATURAL join cannot have qualifier
```

JOIN:

I can also use the JOIN with the accompany USING to join the two tables together. The USING specifies the field that is to be used to link the two tables.

```
SQL> select empno, ename, deptno, dname
2 from emp join dept
3 using (deptno);

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7369 SMITH 20 RESEARCH
7499 ALLEN 30 SALES
7521 WARD 30 SALES
7566 JONES 20 RESEARCH
7654 MARTIN 30 SALES
7698 BLAKE 30 SALES
```

Type here to search

10:18 AM
9/19/2019

File Edit View History Bookmarks Tools Help

JOIN in Oracle:

www.pgrocer.net/Cis50/oraclejoin.html

```
SQL> select empno, ename, emp.deptno, dept.deptno, dname
2 from emp natural join dept;
select empno, ename, emp.deptno, dept.deptno, dname
*
ERROR at line 1:
ORA-25155: column used in NATURAL join cannot have qualifier
```

JOIN:

I can also use the JOIN with the accompany USING to join the two tables together. The USING specifies the field that is to be used to link the two tables.

```
SQL> select empno, ename, deptno, dname
2 from emp join dept
3 using (deptno);

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7369 SMITH 20 RESEARCH
7499 ALLEN 30 SALES
7521 WARD 30 SALES
7566 JONES 20 RESEARCH
7654 MARTIN 30 SALES
7698 BLAKE 30 SALES
7782 CLARK 10 ACCOUNTING
7788 SCOTT 20 RESEARCH
7839 KING 10 ACCOUNTING
7844 TURNER 30 SALES
7876 ADAMS 20 RESEARCH

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7900 JAMES 30 SALES
7902 FORD 20 RESEARCH
7934 MILLER 10 ACCOUNTING

14 rows selected.
```

If the two tables do not have a name in common then you can use the ON clause with the join to specify the relationship to be used in the join. For example if on one table I have used deptnum and on the other table I had used deptno I could make the join as shown.

```
SQL> select empno, ename, emp.deptno, dname
2 from emp join dept
3 on emp.deptno = dept.deptnum;
```

Since on these tables, the names are the same, I will illustrate using this example:

```
SQL> select empno, ename, emp.deptno, dname
2 from emp join dept
3 on emp.deptno = dept.deptno;

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7369 SMITH 20 RESEARCH
7499 ALLEN 30 SALES
7521 WARD 30 SALES
7566 JONES 20 RESEARCH
7654 MARTIN 30 SALES
7698 BLAKE 30 SALES
```

Type here to search

10:19 AM
9/19/2019

Title: Sep 19-10:19 AM (22 of 31)

File Edit View History Bookmarks Tools Help

JOIN in Oracle:

www.pgrocer.net/Cis50/oraclejoin.html

```
7876 ADAMS 20 RESEARCH

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7900 JAMES 30 SALES
7902 FORD 20 RESEARCH
7934 MILLER 10 ACCOUNTING

14 rows selected.
```

If the two tables do not have a name in common then you can use the ON clause with the join to specify the relationship to be used in the join. For example if on one table I have used deptnum and on the other table I had used deptno I could make the join as shown.

```
SQL> select empno, ename, emp.deptno, dname
2 from emp join dept
3 on emp.deptno = dept.deptnum;
```

Since on these tables, the names are the same, I will illustrate using this example:

```
SQL> select empno, ename, emp.deptno, dname
2 from emp join dept
3 on emp.deptno = dept.deptno;

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7369 SMITH 20 RESEARCH
7499 ALLEN 30 SALES
7521 WARD 30 SALES
7566 JONES 20 RESEARCH
7654 MARTIN 30 SALES
7698 BLAKE 30 SALES
7782 CLARK 10 ACCOUNTING
7788 SCOTT 20 RESEARCH
7839 KING 10 ACCOUNTING
7844 TURNER 30 SALES
7876 ADAMS 20 RESEARCH

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7900 JAMES 30 SALES
7902 FORD 20 RESEARCH
7934 MILLER 10 ACCOUNTING

14 rows selected.
```

Outer join:

I am now going to create a new table and add another dept to it. Actually I did not have to do this because looking at emp there were no records with deptno sales. Well, it gave me an opportunity to show you how to create a table from an existing table.

```
SQL> create table deptplus as select * from dept;

Table created.

SQL> select * from deptplus;

DEPTNO DNAME LOC
---------- -------------- -------------
10 ACCOUNTING NEW YORK
20 RESEARCH DALLAS
```

Type here to search

10:22 AM
9/19/2019

Title: Sep 19-10:22 AM (23 of 31)

```
2 values(50, 'IT', 'BOSTON');

1 row created.

SQL> select * from deptplus;

DEPTNO DNAME LOC
---------- -------------- -------------
10 ACCOUNTING NEW YORK
20 RESEARCH DALLAS
30 SALES CHICAGO
40 OPERATIONS BOSTON
50 IT BOSTON
```

I am now going to look at an outer join using these tables: emp and deptplus. Note that the table listed to the right is the table that will show the unmatched items.

```
SQL> select empno, ename, emp.deptno, dname
2 from emp right outer join deptplus
3 on emp.deptno = deptplus.deptno;

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7369 SMITH 20 RESEARCH
7499 ALLEN 30 SALES
7521 WARD 30 SALES
7566 JONES 20 RESEARCH
7654 MARTIN 30 SALES
7698 BLAKE 30 SALES
7782 CLARK 10 ACCOUNTING
7788 SCOTT 20 RESEARCH
7839 KING 10 ACCOUNTING
7844 TURNER 30 SALES
7876 ADAMS 20 RESEARCH

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7900 JAMES 30 SALES
7902 FORD 20 RESEARCH
7934 MILLER 10 ACCOUNTING
IT
OPERATIONS
16 rows selected.
```

When I change the word right to left, I will see different results. The two departments that do not have any employees in them will not be listed.

```
SQL> select empno, ename, emp.deptno, dname
2 from emp left outer join deptplus
3 on emp.deptno = deptplus.deptno;

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7934 MILLER 10 ACCOUNTING
7839 KING 10 ACCOUNTING
7782 CLARK 10 ACCOUNTING
7902 FORD 20 RESEARCH
7876 ADAMS 20 RESEARCH
7788 SCOTT 20 RESEARCH
7566 JONES 20 RESEARCH
7369 SMITH 20 RESEARCH
7900 JAMES 30 SALES
```

File Edit View History Bookmarks Tools Help

JOIN in Oracle:

www.pgrocer.net/Cis50/oraclejoin.html

```
7782 CLARK 10 ACCOUNTING
7902 FORD 20 RESEARCH
7876 ADAMS 20 RESEARCH
7788 SCOTT 20 RESEARCH
7566 JONES 20 RESEARCH
7369 SMITH 20 RESEARCH
7900 JAMES 30 SALES
7844 TURNER 30 SALES
7698 BLAKE 30 SALES

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7654 MARTIN 30 SALES
7521 WARD 30 SALES
7499 ALLEN 30 SALES

14 rows selected.
```

I am now creating a copy of the emp table called empplus and adding a record with deptno 60. Note that deptno 60 does not exist in the dept table. I will then run a left join to show John Doe and dept 60 with no deptno.

```
SQL> create table empplus as select * from emp;

SQL> insert into empplus
2 values (8888,'John Doe','ANALYST',7788,'10-NOV-88',2900,100,60);

1 row created.

SQL> select empno, ename, empplus.deptno, dname
2 from empplus left outer join deptplus
3 on empplus.deptno = deptplus.deptno;

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7934 MILLER 10 ACCOUNTING
7839 KING 10 ACCOUNTING
7782 CLARK 10 ACCOUNTING
7902 FORD 20 RESEARCH
7876 ADAMS 20 RESEARCH
7788 SCOTT 20 RESEARCH
7566 JONES 20 RESEARCH
7369 SMITH 20 RESEARCH
7900 JAMES 30 SALES
7844 TURNER 30 SALES
7698 BLAKE 30 SALES

EMPNO ENAME DEPTNO DNAME
---------- ---------- ---------- --------------
7654 MARTIN 30 SALES
7521 WARD 30 SALES
7499 ALLEN 30 SALES
8888 John Doe 60

15 rows selected.
```

Non-equality or non-equi joins:

Now I am going to run a non-equality join using the salgrade table and come up with the grade for each employee using a non-equality join.

```
SQL> select * from salgrade;
```

Type here to search

10:25 AM
9/19/2019

Non-equality or non-equi joins:

Now I am going to run a non-equality join using the salgrade table and come up with the grade for each employee using a non-equality join.

```
SQL> select * from salgrade;

GRADE LOSAL HISAL
---------- ---------- ----------
1 700 1200
2 1201 1400
3 1401 2000
4 2001 3000
5 3001 9999

SQL> select empno, ename, sal, grade, losal, hisal
2 from emp join salgrade
3 on sal between losal and hisal;

EMPNO ENAME SAL GRADE LOSAL HISAL
---------- ---------- ---------- ---------- ---------- ----------
7369 SMITH 800 1 700 1200
7876 ADAMS 1100 1 700 1200
7900 JAMES 950 1 700 1200
7521 WARD 1250 2 1201 1400
7654 MARTIN 1250 2 1201 1400
7934 MILLER 1300 2 1201 1400
7499 ALLEN 1600 3 1401 2000
7844 TURNER 1500 3 1401 2000
7566 JONES 2975 4 2001 3000
7698 BLAKE 2850 4 2001 3000
7782 CLARK 2450 4 2001 3000

EMPNO ENAME SAL GRADE LOSAL HISAL
---------- ---------- ---------- ---------- ---------- ----------
7788 SCOTT 3000 4 2001 3000
7902 FORD 3000 4 2001 3000
7839 KING 5000 5 3001 9999

14 rows selected.
```

Note: using the where clause this would be written:

```
SQL> select empno, ename, sal, grade, losal, hisal
2 from emp, salgrade
3 where sal between losal and hisal;

EMPNO ENAME SAL GRADE LOSAL HISAL
---------- ---------- ---------- ---------- ---------- ----------
7369 SMITH 800 1 700 1200
7876 ADAMS 1100 1 700 1200
7900 JAMES 950 1 700 1200
7521 WARD 1250 2 1201 1400
7654 MARTIN 1250 2 1201 1400
7934 MILLER 1300 2 1201 1400
7499 ALLEN 1600 3 1401 2000
7844 TURNER 1500 3 1401 2000
7566 JONES 2975 4 2001 3000
7698 BLAKE 2850 4 2001 3000
```

the query on one table are used to extract data from another query. Subqueries can also be based on the same table when you want to select data in the table based on an initial conditional check of the table. The subquery can be located in a FROM, WHERE or HAVING clause and can use the tests of >, <, = , !=, >=, <=, ANY, ALL, IN. The structure of the subquery calls for the inner query which is resolved first to be enclosed in quotes. It is also important to note that the value returned from the inner select is the value that the outer select is testing on - the code must reflect this fact. The examples below show a variety of uses of the select within a select.

In the first example, I want to find out what the year goal is for someone with the last name of Ash. Once I find that out, I want to display everyone that has a year goal greater than the year goal of Ash. Notice the structure: the inner select is retrieving a yrgoal and the outer select is testing against yrgoal.

SQL CODE:

Starting subqueries.

```
SQL> SELECT idno, name, yrgoal, contact
  2  FROM donor
  3  WHERE yrgoal >
  4      (SELECT yrgoal
  5       FROM donor
  6       WHERE name LIKE '%Ash');

IDNO  NAME               YRGOAL CONTACT
----- --------------- --------- ------------
11111 Stephen Daniels       500 John Smith
12121 Jennifer Ames         400 Susan Jones
```

In the second example, the inner select comes up with all of the drive numbers where the contribution amount is greater than 20. The outer select shows information for all rows where the drive number is in the list of drives that resulted from the inner select. It is important to note the use of IN in the outer select. The inner select has a chance of returning more than one value and the outer select has to be able to deal with multiple returns. A good way of handling this is with the IN.

SQL CODE:

```
SQL> SELECT idno, driveno, contdate, contamt
  2  FROM donation
  3  WHERE driveno IN
  4     (SELECT driveno
  5      FROM donation
  6      WHERE contamt > 20);

IDNO  DRI CONTDATE    CONTAMT
----- --- --------- ---------
11111 100 07-JAN-99        25
23456 100 03-MAR-99        20
22222 100 14-MAR-99        10
12121 200 23-FEB-99        40
```

In the third example, the inner select finds the average yrgoal from the donor file and then the outer select finds all donors where the yrgoal is greater than or equal to the average. Again the inner select is returning the average for yrgoal and the outer select is testing against yrgoal.

SQL CODE:

```
SQL> SELECT name, yrgoal
  2  FROM donor
  3  WHERE yrgoal >=
```

File Edit View History Bookmarks Tools Help

Subqueries...

www.pgrocer.net/Cis50/orasubqu.html

SUBQUERIES:

A subquery is a SELECT within a select (also called nested, sub, and inner). When the SELECT is executed, the inner SELECT is executed and the results are used to resolve the outer SELECT. Subqueries can be based on multiple tables where the result of the query on one table are used to extract data from another query. Subqueries can also be based on the same table when you want to select data in the table based on an initial conditional check of the table. The subquery can be located in a FROM, WHERE or HAVING clause and can use the tests of >, <, = , !=, >=, <=, ANY, ALL, IN. The structure of the subquery calls for the inner query which is resolved first to be enclosed in quotes. It is also important to note that the value returned from the inner select is the value that the outer select is testing on - the code must reflect this fact. The examples below show a variety of uses of the select within a select.

In the first example, I want to find out what the year goal is for someone with the last name of Ash. Once I find that out, I want to display everyone that has a year goal greater than the year goal of Ash. Notice the structure: the inner select is retrieving a yrgoal and the outer select is testing against yrgoal.

SQL CODE:

```
SQL> SELECT idno, name, yrgoal, contact
  2  FROM donor
  3  WHERE yrgoal >
  4      (SELECT yrgoal
  5       FROM donor
  6       WHERE name LIKE '%Ash');

IDNO  NAME               YRGOAL CONTACT
----- --------------- --------- ------------
11111 Stephen Daniels       500 John Smith
12121 Jennifer Ames         400 Susan Jones
```

In the second example, the inner select comes up with all of the drive numbers where the contribution amount is greater than 20. The outer select shows information for all rows where the drive number is in the list of drives that resulted from the inner select. It is important to note the use of IN in the outer select. The inner select has a chance of returning more than one value and the outer select has to be able to deal with multiple returns. A good way of handling this is with the IN.

SQL CODE:

```
SQL> SELECT idno, driveno, contdate, contamt
  2  FROM donation
  3  WHERE driveno IN
  4     (SELECT driveno
  5      FROM donation
  6      WHERE contamt > 20);

IDNO  DRI CONTDATE    CONTAMT
----- --- --------- ---------
11111 100 07-JAN-99        25
23456 100 03-MAR-99        20
22222 100 14-MAR-99        10
12121 200 23-FEB-99        40
```

In the third example, the inner select finds the average yrgoal from the donor file and then the outer select finds all donors where the yrgoal is greater than or equal to the average. Again the inner select is returning the average for yrgoal and the outer select is testing against yrgoal.

Type here to search

10:33 AM
9/19/2019

File Edit View History Bookmarks Tools Help

Subqueries...

www.pgrocer.net/Cis50/orasubqu.html

yrgoal and the outer select is testing against yrgoal.

SQL CODE:

```
SQL> SELECT idno, name, yrgoal, contact
  2  FROM donor
  3  WHERE yrgoal >
  4      (SELECT yrgoal
  5       FROM donor
  6       WHERE name LIKE '%Ash');

IDNO  NAME                YRGOAL CONTACT
----- --------------- --------- ------------
11111 Stephen Daniels       500 John Smith
12121 Jennifer Ames         400 Susan Jones
```

one record

In the second example, the inner select comes up with all of the drive numbers where the contribution amount is greater than 20. The outer select shows information for all rows where the drive number is in the list of drives that resulted from the inner select. It is important to note the use of IN in the outer select. The inner select has a chance of returning more than one value and the outer select has to be able to deal with multiple returns. A good way of handling this is with the IN.

SQL CODE:

```
SQL> SELECT idno, driveno, contdate, contamt
  2  FROM donation
  3  WHERE driveno IN
  4     (SELECT driveno
  5      FROM donation
  6      WHERE contamt > 20);

IDNO  DRI CONTDATE     CONTAMT
----- --- --------- ---------
11111 100 07-JAN-99        25
23456 100 03-MAR-99        20
22222 100 14-MAR-99        10
12121 200 23-FEB-99        40
```

multiple possible

In the third example, the inner select finds the average yrgoal from the donor file and then the outer select finds all donors where the yrgoal is greater than or equal to the average. Again the inner select is returning the average for yrgoal and the outer select is testing against yrgoal.

SQL CODE:

```
SQL> SELECT name, yrgoal
  2  FROM donor
  3  WHERE yrgoal >=
  4     (SELECT AVG(yrgoal)
  5      FROM donor);

NAME               YRGOAL
--------------- ---------
Stephen Daniels       500
Jennifer Ames         400
```

Type here to search

10:35 AM
9/19/2019

File Edit View History Bookmarks Tools Help

Subqueries...

www.pgrocer.net/Cis50/orasubqu.html

SUBQUERIES:

A subquery is a SELECT within a select (also called nested, sub, and inner). When th
the query on one table are used to extract data from another query. Subqueries can al
or HAVING clause and can use the tests of >, <, = , !=, >=, <=, ANY, ALL, IN. The s
the value that the outer select is testing on - the code must reflect this fact. The examp

In the first example, I want to find out what the year goal is for someone with the last
yrgoal and the outer select is testing against yrgoal.

SQL CODE:

```
SQL> SELECT idno, name, yrgoal, contact
  2  FROM donor
  3  WHERE yrgoal >
  4       (SELECT yrgoal
  5        FROM donor
  6        WHERE name LIKE '%Ash');

IDNO  NAME                YRGOAL CONTACT
----- --------------- --------- ------------
11111 Stephen Daniels       500 John Smith
12121 Jennifer Ames         400 Susan Jones
```

In the second example, the inner select comes up with all of the drive numbers where
select. It is important to note the use of IN in the outer select. The inner select has a c

SQL CODE:

```
SQL> SELECT idno, driveno, contdate, contamt
  2  FROM donation
  3  WHERE driveno IN
  4      (SELECT driveno
  5       FROM donation
  6       WHERE contamt > 20);

IDNO  DRI CONTDATE    CONTAMT
----- --- --------- ---------
11111 100 07-JAN-99        25
23456 100 03-MAR-99        20
22222 100 14-MAR-99        10
12121 200 23-FEB-99        40
```

In the third example, the inner select finds the average yrgoal from the donor file and then the outer select finds all donors where the yrgoal is greater than or equal to the average. Again the inner select is returning the average for yrgoal and the outer select is testing against yrgoal.

*Untitled - Notepad

File Edit Format View Help

```
SQL> SELECT * from donor;

IDNO  NAME            STADR           CITY       ST ZIP   DATEFST      YRGOAL CONTACT
----- --------------- --------------- ---------- -- ----- --------- --------- ------------
11111 Stephen Daniels 123 Elm St      Seekonk    MA 02345 03-JUL-98       500 John Smith
12121 Jennifer Ames   24 Benefit St   Providence RI 02045 24-MAY-97       400 Susan Jones
22222 Carl Hersey     24 Benefit St   Providence RI 02045 03-JAN-98           Susan Jones
23456 Susan Ash       21 Main St      Fall River MA 02720 04-MAR-92       100 Amy Costa
33333 Nancy Taylor    26 Oak St       Fall River MA 02720 04-MAR-92        50 John Adams
34567 Robert Brooks   36 Pine St      Fall River MA 02720 04-APR-98        50 Amy Costa

IDNO  DRI CONTDATE    CONTAMT
----- --- --------- ---------
11111 100 07-JAN-99        25
12121 200 23-FEB-99        40
23456 100 03-MAR-99        20
33333 300 10-MAR-99        10

DRI DRIVENAME       DRIVECHAIR    LASTYEAR  THISYEAR
--- --------------- ------------ --------- ---------
100 Kids Shelter    Ann Smith        10000         0
200 Animal Home     Linda Grant       5000         0
300 Health Aid      David Ross        7000         0
```

This data is different from the data on the handout.

Ln 23, Col 53 100% Windows (CRLF) UTF

Type here to search

10:38 AM 9/19/2019

File Edit View History Bookmarks Tools Help

Subqueries...

www.pgrocer.net/Cis50/orasubqu.html

```
11111 100 07-JAN-99        25
23456 100 03-MAR-99        20
22222 100 14-MAR-99        10
12121 200 23-FEB-99        40
```

In the third example, the inner select finds the average yrgoal from the donor file and then the outer select finds all donors where the yrgoal is greater than or equal to the average. Again the inner select is returning the average for yrgoal and the outer select is testing against yrgoal.

SQL CODE:

```
SQL> SELECT name, yrgoal
  2  FROM donor
  3  WHERE yrgoal >=
  4     (SELECT AVG(yrgoal)
  5      FROM donor);

NAME               YRGOAL
--------------- ---------
Stephen Daniels       500
Jennifer Ames         400
```

One thing

In the fourth example, I am using two tables. I am using the inner select to look at the donation table and select all idno where the contamt is greater than 20. The outer select than prints the name and yrgoal from the donor table for those donors that were selected by the inner select. Again, the inner select is coming up with one or more idno and the outer select is testing for a idno in the list that was retrieved.

SQL CODE:

```
1  SELECT name, yrgoal
  2  FROM donor
  3  WHERE idno IN
  4     (SELECT idno
  5      FROM donation
  6*     WHERE contamt > 20)
SQL> /

NAME               YRGOAL
--------------- ---------
Stephen Daniels       500
Jennifer Ames         400
```

could be multiple

In the fifth example, I am only using the donor table. In the inner select I am finding the average of yrgoal and then in the outer select I am showing the groups by state and contact within state where the sum of the yrgoal is greater than the average yrgoal.

SQL CODE:

```
SQL> SELECT state, contact, SUM(yrgoal)
  2  FROM donor
  3  GROUP BY state, contact
  4  HAVING SUM(yrgoal) >
  5     (SELECT AVG(yrgoal)
  6      FROM donor);

ST CONTACT        SUM(YRGOAL)
```

Type here to search

10:41 AM 9/19/2019

Title: Sep 19-10:41 AM (31 of 31)