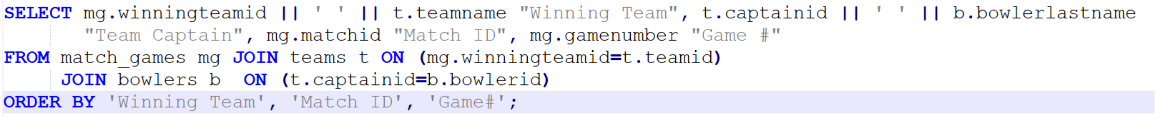
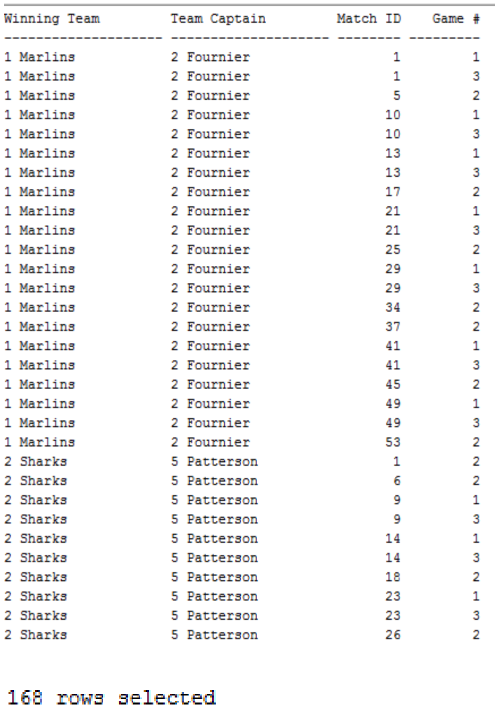
1. Write a SELECT statement to display only <u>four</u> columns – all winning teams, team captains, match id, and game numbers that they won. Do not display teams if they did not win. Sort the results by the winning team, match id, and game number in ascending order. Use proper column headings. You must use table join. See sample output below.



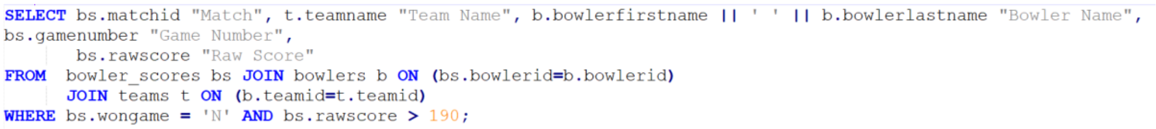
```
SELECT mg.winningteamid || ' ' || t.teamname "Winning Team", t.captainid || ' ' || b.bowlerlastname
       "Team Captain", mg.matchid "Match ID", mg.gamenumber "Game #"
FROM match_games mg JOIN teams t ON (mg.winningteamid=t.teamid)
     JOIN bowlers b  ON (t.captainid=b.bowlerid)
ORDER BY 'Winning Team', 'Match ID', 'Game#';
```



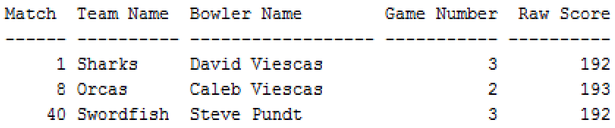
```
Winning Team         Team Captain         Match ID    Game #
-------------------- -------------------- -------- ---------
1 Marlins            2 Fournier                  1         1
1 Marlins            2 Fournier                  1         3
1 Marlins            2 Fournier                  5         2
1 Marlins            2 Fournier                 10         1
1 Marlins            2 Fournier                 10         3
1 Marlins            2 Fournier                 13         1
1 Marlins            2 Fournier                 13         3
1 Marlins            2 Fournier                 17         2
1 Marlins            2 Fournier                 21         1
1 Marlins            2 Fournier                 21         3
1 Marlins            2 Fournier                 25         2
1 Marlins            2 Fournier                 29         1
1 Marlins            2 Fournier                 29         3
1 Marlins            2 Fournier                 34         2
1 Marlins            2 Fournier                 37         2
1 Marlins            2 Fournier                 41         1
1 Marlins            2 Fournier                 41         3
1 Marlins            2 Fournier                 45         2
1 Marlins            2 Fournier                 49         1
1 Marlins            2 Fournier                 49         3
1 Marlins            2 Fournier                 53         2
2 Sharks             5 Patterson                 1         2
2 Sharks             5 Patterson                 6         2
2 Sharks             5 Patterson                 9         1
2 Sharks             5 Patterson                 9         3
2 Sharks             5 Patterson                14         1
2 Sharks             5 Patterson                14         3
2 Sharks             5 Patterson                18         2
2 Sharks             5 Patterson                23         1
2 Sharks             5 Patterson                23         3
2 Sharks             5 Patterson                26         2

168 rows selected
```

2. Write a SELECT statement that displays only <u>five</u> columns –the match ID, the team name, bowler's first and last name, game number they played in and the bowler's game raw score. The list must <u>only</u> include bowlers with the games they played but did <u>not</u> win and the raw score higher than 190. Use proper column headings. You must use table join. See the output below.

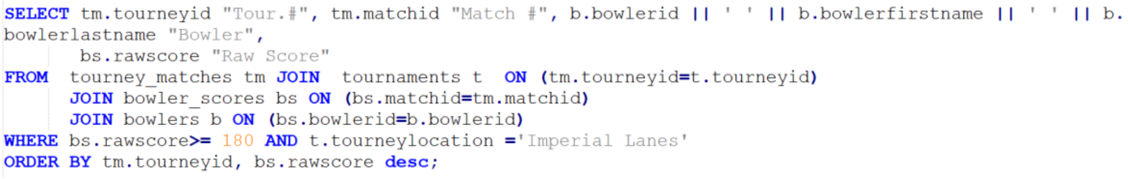


```
SELECT bs.matchid "Match", t.teamname "Team Name", b.bowlerfirstname || ' ' || b.bowlerlastname "Bowler Name",
bs.gamenumber "Game Number",
      bs.rawscore "Raw Score"
FROM  bowler_scores bs JOIN bowlers b ON (bs.bowlerid=b.bowlerid)
      JOIN teams t ON (b.teamid=t.teamid)
WHERE bs.wongame = 'N' AND bs.rawscore > 190;
```

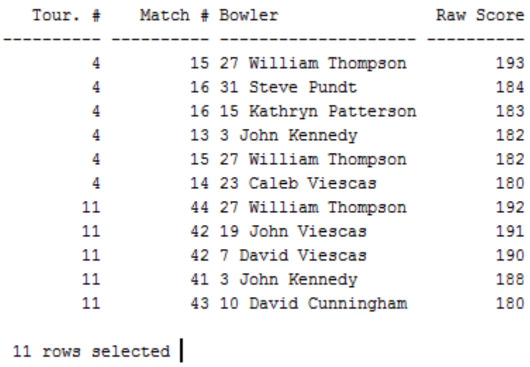


```
Match  Team Name  Bowler Name        Game Number  Raw Score
------ ---------- ------------------ ----------- ----------
     1 Sharks     David Viescas                3        192
     8 Orcas      Caleb Viescas                2        193
    40 Swordfish  Steve Pundt                  3        192
```

3. Write a SELECT statement that displays four columns – tournament ID, match ID, bowler id, first name, last name, and raw score. You must use table join. Include the bowlers who had a raw score of 180 or better at Imperial Lanes only. Use proper column headings. Sort the results by tournament ID first, then raw score in descending order. See sample output below.



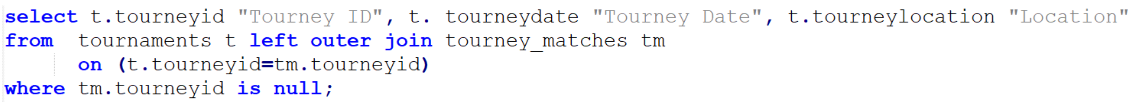
```
SELECT tm.tourneyid "Tour.#", tm.matchid "Match #", b.bowlerid || ' ' || b.bowlerfirstname || ' ' || b.
bowlerlastname "Bowler",
       bs.rawscore "Raw Score"
FROM   tourney_matches tm JOIN  tournaments t  ON (tm.tourneyid=t.tourneyid)
       JOIN bowler_scores bs ON (bs.matchid=tm.matchid)
       JOIN bowlers b ON (bs.bowlerid=b.bowlerid)
WHERE bs.rawscore>= 180 AND t.tourneylocation ='Imperial Lanes'
ORDER BY tm.tourneyid, bs.rawscore desc;
```



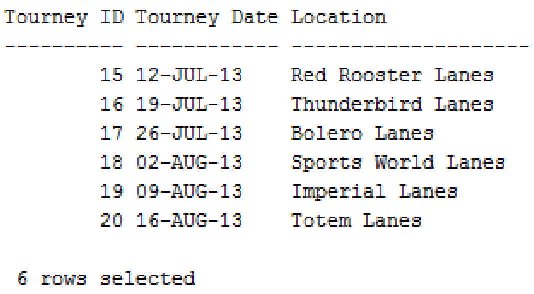
```
  Tour. #    Match # Bowler               Raw Score
---------- ---------- -------------------- ----------
         4         15 27 William Thompson         193
         4         16 31 Steve Pundt              184
         4         16 15 Kathryn Patterson        183
         4         13 3 John Kennedy              182
         4         15 27 William Thompson         182
         4         14 23 Caleb Viescas            180
        11         44 27 William Thompson         192
        11         42 19 John Viescas             191
        11         42 7 David Viescas             190
        11         41 3 John Kennedy              188
        11         43 10 David Cunningham         180

 11 rows selected
```

4. Write a SELECT statement that displays tournaments that have not been played yet. You must join tables. Use proper column headings. See sample output below.



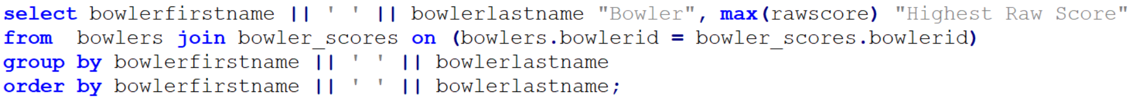
```
select t.tourneyid "Tourney ID", t. tourneydate "Tourney Date", t.tourneylocation "Location"
from   tournaments t left outer join tourney_matches tm
       on (t.tourneyid=tm.tourneyid)
where tm.tourneyid is null;
```



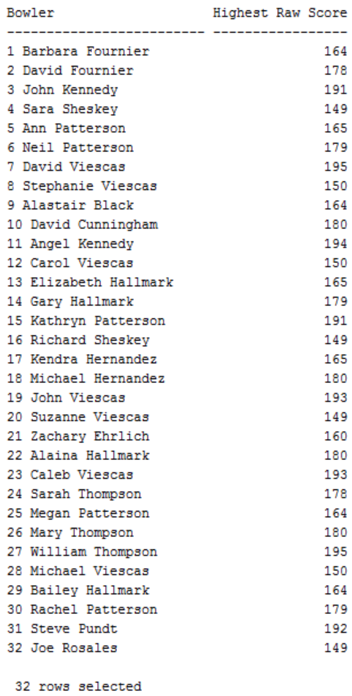
```
Tourney ID Tourney Date Location
---------- ------------ --------------------
        15 12-JUL-13    Red Rooster Lanes
        16 19-JUL-13    Thunderbird Lanes
        17 26-JUL-13    Bolero Lanes
        18 02-AUG-13    Sports World Lanes
        19 09-AUG-13    Imperial Lanes
        20 16-AUG-13    Totem Lanes

 6 rows selected
```

5. Write a SELECT statement that displays two columns – bowlers and highest raw score for each bowler. Use proper column headings. Sort the results by bowler in ascending order. See sample output below.



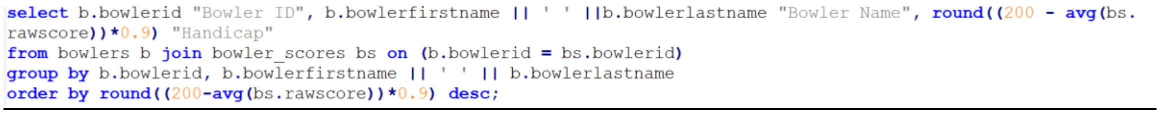
```
select bowlerfirstname || ' ' || bowlerlastname "Bowler", max(rawscore) "Highest Raw Score"
from  bowlers join bowler_scores on (bowlers.bowlerid = bowler_scores.bowlerid)
group by bowlerfirstname || ' ' || bowlerlastname
order by bowlerfirstname || ' ' || bowlerlastname;
```



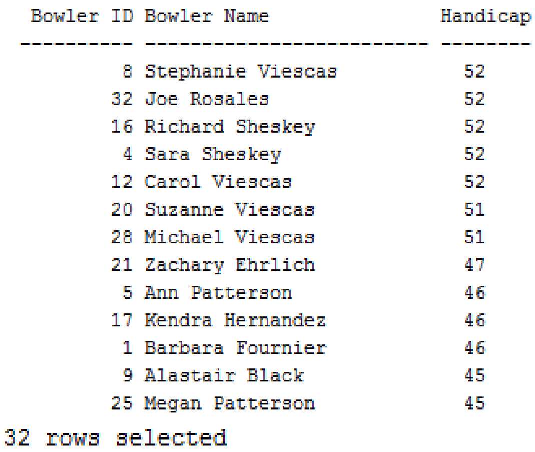
```
Bowler                   Highest Raw Score
------------------------ -----------------
1 Barbara Fournier                     164
2 David Fournier                       178
3 John Kennedy                         191
4 Sara Sheskey                         149
5 Ann Patterson                        165
6 Neil Patterson                       179
7 David Viescas                        195
8 Stephanie Viescas                    150
9 Alastair Black                       164
10 David Cunningham                    180
11 Angel Kennedy                       194
12 Carol Viescas                       150
13 Elizabeth Hallmark                  165
14 Gary Hallmark                       179
15 Kathryn Patterson                   191
16 Richard Sheskey                     149
17 Kendra Hernandez                    165
18 Michael Hernandez                   180
19 John Viescas                        193
20 Suzanne Viescas                     149
21 Zachary Ehrlich                     160
22 Alaina Hallmark                     180
23 Caleb Viescas                       193
24 Sarah Thompson                      178
25 Megan Patterson                     164
26 Mary Thompson                       180
27 William Thompson                    195
28 Michael Viescas                     150
29 Bailey Hallmark                     164
30 Rachel Patterson                    179
31 Steve Pundt                         192
32 Joe Rosales                         149

 32 rows selected
```

6. Write a SELECT statement to display the handicap held by bowlers. Assume that the basis score is 200 and the percentage factor is 90%. To calculate the handicap, subtract the average raw score from the basis score and multiply the result by the percentage factor. Round the average raw score in the calculation. Display the results without any decimal points. Sort the results by handicap in descending order. Use proper column headings. See sample output below.

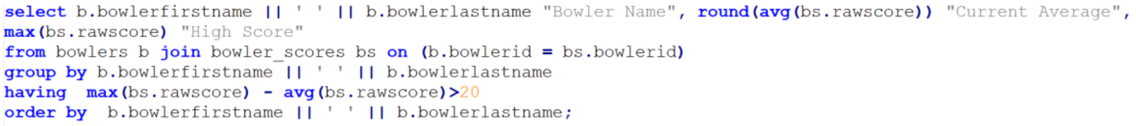


```
select b.bowlerid "Bowler ID", b.bowlerfirstname || ' ' ||b.bowlerlastname "Bowler Name", round((200 - avg(bs.
rawscore))*0.9) "Handicap"
from bowlers b join bowler_scores bs on (b.bowlerid = bs.bowlerid)
group by b.bowlerid, b.bowlerfirstname || ' ' || b.bowlerlastname
order by round((200-avg(bs.rawscore))*0.9) desc;
```

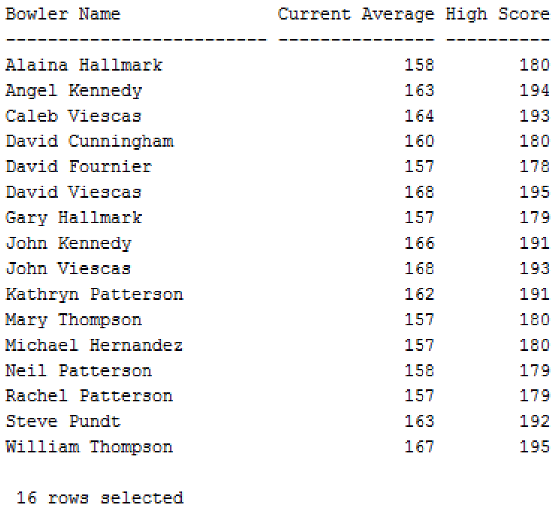


```
 Bowler ID Bowler Name               Handicap
---------- ------------------------- --------
         8 Stephanie Viescas           52
        32 Joe Rosales                 52
        16 Richard Sheskey             52
         4 Sara Sheskey                52
        12 Carol Viescas               52
        20 Suzanne Viescas             51
        28 Michael Viescas             51
        21 Zachary Ehrlich             47
         5 Ann Patterson               46
        17 Kendra Hernandez            46
         1 Barbara Fournier            46
         9 Alastair Black              45
        25 Megan Patterson             45
32 rows selected
```

7. Write a SELECT statement to display the bowlers whose highest raw scores are more than 20 pins higher than their current average raw scores. Use proper column headings. Sort the results by bowler name. See sample output below.



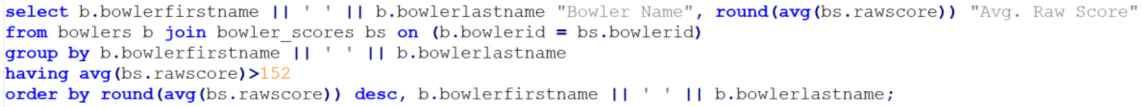
```
select b.bowlerfirstname || ' ' || b.bowlerlastname "Bowler Name", round(avg(bs.rawscore)) "Current Average",
max(bs.rawscore) "High Score"
from bowlers b join bowler_scores bs on (b.bowlerid = bs.bowlerid)
group by b.bowlerfirstname || ' ' || b.bowlerlastname
having  max(bs.rawscore) - avg(bs.rawscore)>20
order by  b.bowlerfirstname || ' ' || b.bowlerlastname;
```



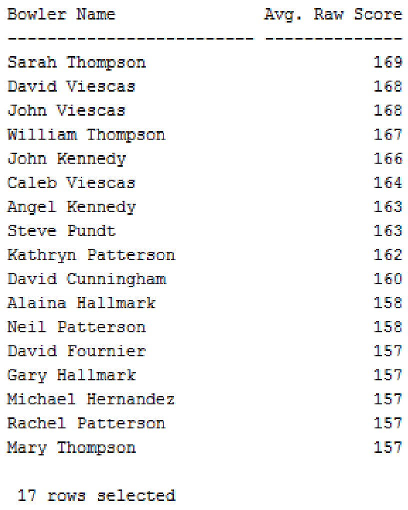
```
Bowler Name               Current Average High Score
------------------------- --------------- ----------
Alaina Hallmark                       158        180
Angel Kennedy                         163        194
Caleb Viescas                         164        193
David Cunningham                      160        180
David Fournier                        157        178
David Viescas                         168        195
Gary Hallmark                         157        179
John Kennedy                          166        191
John Viescas                          168        193
Kathryn Patterson                     162        191
Mary Thompson                         157        180
Michael Hernandez                     157        180
Neil Patterson                        158        179
Rachel Patterson                      157        179
Steve Pundt                           163        192
William Thompson                      167        195

 16 rows selected
```

8. Write a SELECT statement to display the bowler name and the average of the bowler's raw scores for bowlers whose average is greater than 152. Round the average scores. Use proper column headings. Sort the results by the average scores in descending order and bowler last name and bowler first name. See sample output below.



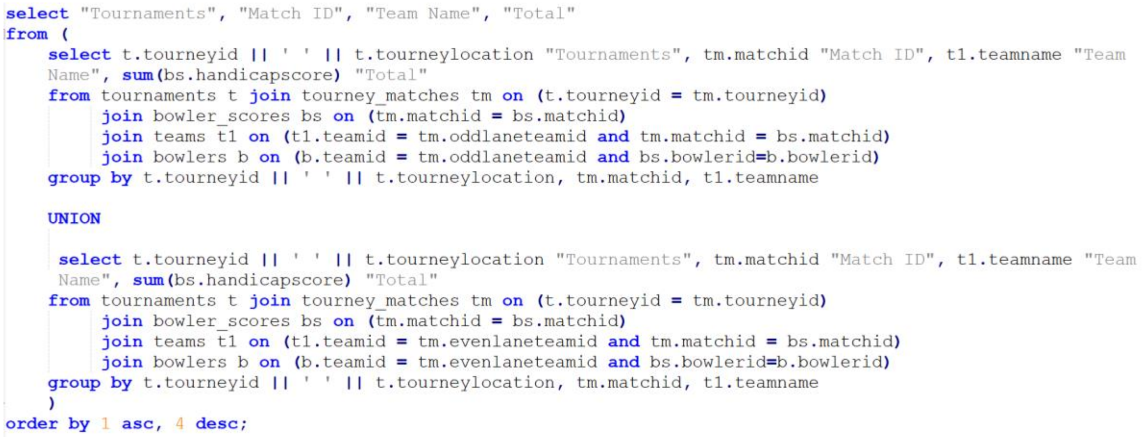
```
select b.bowlerfirstname || ' ' || b.bowlerlastname "Bowler Name", round(avg(bs.rawscore)) "Avg. Raw Score"
from bowlers b join bowler_scores bs on (b.bowlerid = bs.bowlerid)
group by b.bowlerfirstname || ' ' || b.bowlerlastname
having avg(bs.rawscore)>152
order by round(avg(bs.rawscore)) desc, b.bowlerfirstname || ' ' || b.bowlerlastname;
```



```
Bowler Name               Avg. Raw Score
------------------------- --------------
Sarah Thompson                       169
David Viescas                        168
John Viescas                         168
William Thompson                     167
John Kennedy                         166
Caleb Viescas                        164
Angel Kennedy                        163
Steve Pundt                          163
Kathryn Patterson                    162
David Cunningham                     160
Alaina Hallmark                      158
Neil Patterson                       158
David Fournier                       157
Gary Hallmark                        157
Michael Hernandez                    157
Rachel Patterson                     157
Mary Thompson                        157

 17 rows selected
```

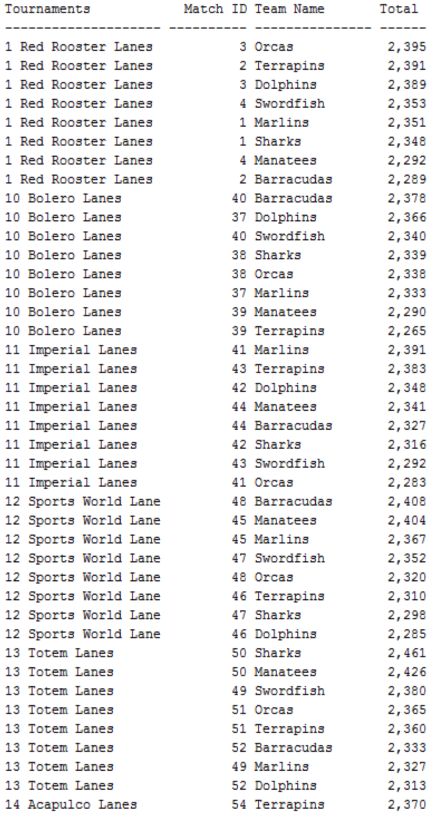
9. Write a SELECT statement to display each tournament id and name, the tournament location, match ID, the name of the each team, and the total of the handicap score for each team. Use proper column headings and formatting. Sort the results by tournament ID in ascending order and the total handicap score in descending order. See sample output below.



```
select "Tournaments", "Match ID", "Team Name", "Total"
from (
    select t.tourneyid || ' ' || t.tourneylocation "Tournaments", tm.matchid "Match ID", t1.teamname "Team
    Name", sum(bs.handicapscore) "Total"
    from tournaments t join tourney_matches tm on (t.tourneyid = tm.tourneyid)
         join bowler_scores bs on (tm.matchid = bs.matchid)
         join teams t1 on (t1.teamid = tm.oddlaneteamid and tm.matchid = bs.matchid)
         join bowlers b on (b.teamid = tm.oddlaneteamid and bs.bowlerid=b.bowlerid)
    group by t.tourneyid || ' ' || t.tourneylocation, tm.matchid, t1.teamname

    UNION

     select t.tourneyid || ' ' || t.tourneylocation "Tournaments", tm.matchid "Match ID", t1.teamname "Team
     Name", sum(bs.handicapscore) "Total"
    from tournaments t join tourney_matches tm on (t.tourneyid = tm.tourneyid)
         join bowler_scores bs on (tm.matchid = bs.matchid)
         join teams t1 on (t1.teamid = tm.evenlaneteamid and tm.matchid = bs.matchid)
         join bowlers b on (b.teamid = tm.evenlaneteamid and bs.bowlerid=b.bowlerid)
    group by t.tourneyid || ' ' || t.tourneylocation, tm.matchid, t1.teamname
    )
order by 1 asc, 4 desc;
```

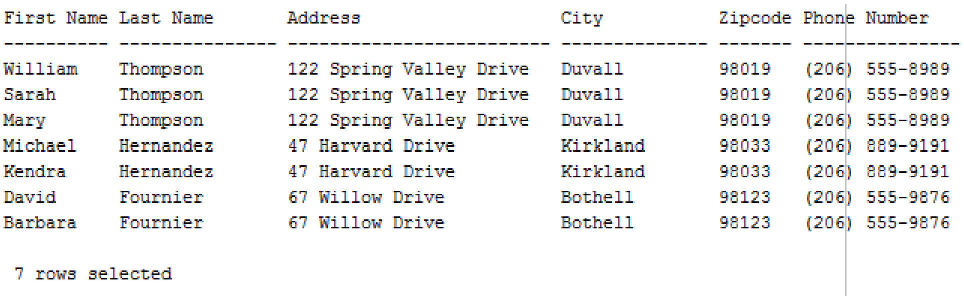


```
Tournaments          Match ID Team Name        Total
-------------------- ---------- ---------------- ------
1 Red Rooster Lanes           3 Orcas           2,395
1 Red Rooster Lanes           2 Terrapins       2,391
1 Red Rooster Lanes           3 Dolphins        2,389
1 Red Rooster Lanes           4 Swordfish       2,353
1 Red Rooster Lanes           1 Marlins         2,351
1 Red Rooster Lanes           1 Sharks          2,348
1 Red Rooster Lanes           4 Manatees        2,292
1 Red Rooster Lanes           2 Barracudas      2,289
10 Bolero Lanes              40 Barracudas      2,378
10 Bolero Lanes              37 Dolphins        2,366
10 Bolero Lanes              40 Swordfish       2,340
10 Bolero Lanes              38 Sharks          2,339
10 Bolero Lanes              38 Orcas           2,338
10 Bolero Lanes              37 Marlins         2,333
10 Bolero Lanes              39 Manatees        2,290
10 Bolero Lanes              39 Terrapins       2,265
11 Imperial Lanes            41 Marlins         2,391
11 Imperial Lanes            43 Terrapins       2,383
11 Imperial Lanes            42 Dolphins        2,348
11 Imperial Lanes            44 Manatees        2,341
11 Imperial Lanes            44 Barracudas      2,327
11 Imperial Lanes            42 Sharks          2,316
11 Imperial Lanes            43 Swordfish       2,292
11 Imperial Lanes            41 Orcas           2,283
12 Sports World Lane         48 Barracudas      2,408
12 Sports World Lane         45 Manatees        2,404
12 Sports World Lane         45 Marlins         2,367
12 Sports World Lane         47 Swordfish       2,352
12 Sports World Lane         48 Orcas           2,320
12 Sports World Lane         46 Terrapins       2,310
12 Sports World Lane         47 Sharks          2,298
12 Sports World Lane         46 Dolphins        2,285
13 Totem Lanes               50 Sharks          2,461
13 Totem Lanes               50 Manatees        2,426
13 Totem Lanes               49 Swordfish       2,380
13 Totem Lanes               51 Orcas           2,365
13 Totem Lanes               51 Terrapins       2,360
13 Totem Lanes               52 Barracudas      2,333
13 Totem Lanes               49 Marlins         2,327
13 Totem Lanes               52 Dolphins        2,313
14 Acapulco Lanes            54 Terrapins       2,370
```

```
112 rows selected
```

10. Write a SELECT statement that returns all bowlers whose address contains "Willow" or "Drive" in anywhere. Sort the list in descending order by the last name and first name. Use proper column headings. See the sample output below.

```
select bowlerfirstname "First Name", bowlerlastname "Last Name", bowleraddress "Address", bowlercity "City",
bowlerzip "Zipcode", bowlerphonenumber "Phone Number"
from bowlers
where bowleraddress like '%Willow%' or bowleraddress like '%Drive%'
order by bowlerlastname desc, bowlerfirstname desc;
```



```
First Name Last Name       Address                   City           Zipcode Phone Number
---------- --------------- ------------------------- -------------- ------- --------------
William    Thompson        122 Spring Valley Drive   Duvall         98019   (206) 555-8989
Sarah      Thompson        122 Spring Valley Drive   Duvall         98019   (206) 555-8989
Mary       Thompson        122 Spring Valley Drive   Duvall         98019   (206) 555-8989
Michael    Hernandez       47 Harvard Drive          Kirkland       98033   (206) 889-9191
Kendra     Hernandez       47 Harvard Drive          Kirkland       98033   (206) 889-9191
David      Fournier        67 Willow Drive           Bothell        98123   (206) 555-9876
Barbara    Fournier        67 Willow Drive           Bothell        98123   (206) 555-9876

 7 rows selected
```