Increasingly people are expecting answers to more complex questions based on the data in their databases. This includes tasks such as ranking, calculating moving averages and rolling up data. The queries at first might seem difficult but these are commonly needed business tasks.

Some of the analytical techniques we have used so far include

- the set functions- Count, Max, Min, Sum, Avg - which work on a collection of rows as a unit.
- Group By and Having- which also produce or work with collections of rows as a unit

To actually analyze our data we need more analytical techniques.

First some examples. Suppose we have a table of student grades for each assignment. Each student has 10 assignment scores and we have three students. SQL for this is in the demo, but not listed here. Here I just want you to see what type of results we might get from the data and the general name for that type of query. I do not expect you to figure out how that code work- yet.

We can list the data sorted by the student ID and the assignment numbers. OK that is all the data but it is not information in terms of looking at the data at a somewhat higher level.

```
STU       ASGN      SCORE
--- ---------- ----------
101          1         50
101          2         50
101          3         40
101          4         45
101          5         40
101          6         47
101          7         45
101          8         30
101          9         45
101         10         48
201          1          0
201          2         50
201          3          0
201          4          0
201          5          5
201          6         30
201          7         35
201          8         30
201          9         37
201         10         30
301          1         50
301          2         50
301          3         45
301          4         45
301          5         40
301          6         40
301          7         42
301          8         30
301          9         30
301         10         30

30 rows selected.
```

(**Ranking**) Suppose we want to show the top 3 assignment scores for each student. Note the scores for student 301. This student has 2 scores at 50 and then 2 at 45; since the 3and 4th score are tied for the third place, both of these are shown.

```
STU      SCORE       ASGN
--- ---------- ----------
```

```
101         50          1
101         50          2
101         48         10
201         50          2
201         37          9
201         35          7
301         50          1
301         50          2
301         45          3
301         45          4
```

(**Rollup**) We could show students and their total and average scores.

```
STU   TOTSCORE   AVGSCORE
--- ---------- ----------
101        440         44
201        217       21.7
301        402       40.2
```

But we can also include the aggregates for all of the students using a rollup. The total scores for all students is not terrible useful so I hide it.

```
STUDENT      TOTSCORE AVGSCORE
------------ -------- ---------
101               440     44.00
201               217     21.70
301               402     40.20
All students              35.30
```

We could add the details to that report

```
STUDENT      ASGN  SCORE
------------ ----- -----
101          1        50
101          2        50
101          3        40
101          4        45
101          5        40
101          6        47
101          7        45
101          8        30
101          9        45
101          10       48
101          AVG      44
201          1         0
201          2        50
201          3         0
201          4         0
201          5         5
201          6        30
201          7        35
201          8        30
201          9        37
201          10       30
201          AVG      22
301          1        50
301          2        50
```

```
301          3        45
301          4        45
301          5        40
301          6        40
301          7        42
301          8        30
301          9        30
301          10       30
301          AVG      40
All students AVG      35
```

```
STUDENT      ASGN  Score/Av
------------ ----- --------
101          1        50
101          2        50
101          3        40
101          4        45
101          5        40
101          6        47
101          7        45
101          8        30
101          9        45
101          10       48
101          AVG      44.00
201          1         0
201          2        50
201          3         0
201          4         0
201          5         5
201          6        30
201          7        35
201          8        30
201          9        37
201          10       30
201          AVG      21.70
301          1        50
301          2        50
301          3        45
301          4        45
301          5        40
301          6        40
301          7        42
301          8        30
301          9        30
301          10       30
301          AVG      40.20
All students AVG      35.30

34 rows selected.
```

(**Running Total**) Maybe you want to see the scores for a student but also see the running total of their scores. This is just student 101.

```
      ASGN      SCORE RUNNINGTOTAL
---------- ---------- ------------
```

```
         1         50          50
         2         50         100
         3         40         140
         4         45         185
         5         40         225
         6         47         272
         7         45         317
         8         30         347
         9         45         392
        10         48         440
```

(**Running Total**) Maybe you want to see the scores for each student but also see the running total of their scores.. We need to start the running total over for each student.

```
StID      Asgn#      Score RunningTotal
---- ---------- ---------- ------------
 101          1         50           50
 101          2         50          100
 101          3         40          140
 101          4         45          185
 101          5         40          225
 101          6         47          272
 101          7         45          317
 101          8         30          347
 101          9         45          392
 101         10         48          440
 201          1          0            0
 201          2         50           50
 201          3          0           50
 201          4          0           50
 201          5          5           55
 201          6         30           85
 201          7         35          120
 201          8         30          150
 201          9         37          187
 201         10         30          217
 301          1         50           50
 301          2         50          100
 301          3         45          145
 301          4         45          190
 301          5         40          230
 301          6         40          270
 301          7         42          312
 301          8         30          342
 301          9         30          372
 301         10         30          402
```

Sometimes these problems were solved by using a programming approach or very complex SQL. The various dbms have been adding more features and functions to SQL to help solve these problems. When these techniques are built into the dbms, they can be optimized by the optimizer and using these functions is generally more efficient than other approaches

Many of these tasks involve the use of analytical functions which can be as simple as the Avg. Sum and Count functions we have been using. We can use these functions to calculate aggregate values over a group of rows.

Many of these techniques use a windowing clause and ranking. Windowing uses an Order By clause and might also partition the data into groups based on a criteria such as department id or by year hired. These subsets of data are called window partitions.

Once the data is sorted and possibly partitioned you can use ranking functions to number the rows in the partitions.

Oracle has added a number of functions that do fairly complex calculations; we will use some of these.

One thing that we can do with these techniques is display both detail and aggregated data in the same query. You have done some of this already with subqueries..

These results of these functions are more meaningful with large amounts of data, but to keep the examples simple, we will use small tables. Although these queries might seem complex at first, they are important for analyzing large amounts of data. This unit's material may seem overwhelming but I think you can handle working out the tasks for the assignment.

The demo file for this document also includes the sql for two tables you need to create and populate.

adv_employees

adv_sales The adv_sales table which will have one row for each day's sales for a range of dates.