MICROSOFT ACCESS 2010 SOLUTIONS TO "SQL FUNDAMENTALS – 3RD EDITION" BY PATRICK

by Ke-Wei Ma

keweima@gmail.com

CHAPTER 2 - GETTING INFORMATION FROM A TABLE

SECTION 2-4, PAGE 41

```
select description as food_item,  price from l_foods;
```

SECTION 2-5, PAGE 44

```
select * from l_foods;
```

SECTION 2-6, PAGE 48

```
select distinct supplier_id  from l_foods;
```

SECTION 2-7, PAGE 50

```
select distinct a, b from sec0207;
```

SECTION 2-9, PAGE 56

```
select first_name, last_name
     from l_employees
     where first_name = 'Nancy';
```

SECTION 2-10, PAGE 58

```
select first_name, last_name
     from l_employees
     where employee_id >= 205;
```

SECTION 2-11, PAGE 60

```
select first_name, last_name
     from l_employees
     where employee_id <> 205;

select first_name, last_name
     from l_employees
     where not employee_id = 205;
```

```
select first_name, last_name
     from l_employees
     where not (employee_id = 205);
```

SECTION 2-12, PAGE 63

```
select first_name, last_name, dept_code
     from l_employees
     where dept_code in ('sal','shp','act');
```

SECTION 2-13, PAGE 65

```
select employee_id, first_name, last_name
   from l_employees where employee_id between 201 and 205;
```

SECTION 2-14, PAGE 68

```
select employee_id, first_name, last_name
   from l_employees
   where employee_id like '*1*';
```

SECTION 2-15, PAGE 71

```
select *
   from l_employees
   where manager_id is null;
```

SECTION 2-17, PAGE 74

```
select department_name as dept
   from l_departments;
```

SECTION 2-18, PAGE 76

```
select a, b
   from sec0218
   order by a, b;
```

```
select a, b
   from sec0218
   order by b, a;
```

SECTION 2-19, PAGE 78

```
select a, b, c
   from sec0219
   order by a, b desc, c desc;
```

CHAPTER 3 - COMPOUND CONDITIONS IN THE WHERE CLAUSE

SECTION 3-1, PAGE 87

```
select *
   from l_foods
   where price < 1.00 or price > 5.00;
```

SECTION 3-2, PAGE 90

```
select *
   from l_foods
   where not (price is null);
```

SECTION 3-3, PAGE 92

Be careful with the NOT operator: it binds tight so you need to use parenthesis in order to make the clause work properly.

```
select *
   from l_employees
   where not ((first_name = 'JIM')
          and (last_name = 'BROWN'))
       or
         not ((first_name = 'JIM')
          and (last_name = 'SMITH'))
       or
         not ((first_name = 'DAN')
          and (last_name = 'BROWN'))
       or
         not ((first_name = 'DAN')
          and (last_name = 'SMITH'));
```

SECTION 3-4, PAGE 95

```
select *
     from l_foods
     where (description = 'FRESH SALAD'
               or description = 'SANDWICH'
               or description = 'DESSERT')
          and (price <= 2.50
               and price_increase <= 0.25);
```

SECTION 3-6, PAGE 101

```
select min_price, max_price, description
   from l_foods, sec0306_price_constants
   where price between min_price and max_price
   order by description;
```

SECTION 3-7, PAGE 105

```
select *
   from l_suppliers
   where supplier_name = 'frank reed''s vegetables';

select *
   from l_suppliers
   where supplier_name = "frank reed's vegetables";
```

SECTION 3-15, PAGE 124

NOT A CODING EXERCISE

CHAPTER 4 – SAVING YOUR RESULTS

SECTION 4-1, PAGE 139

```
select *
   into sec0401_employees
   from l_employees;
```

SECTION 4-2, PAGE 141

```
select *
   from l_employees;
```

Run this command, save query (hit Ctrl+S) and then provide a “view” name.

SECTION 4-5, PAGE 144

NOT A CODING EXERCISE

SECTION 4-6, PAGE 145

NOT A CODING EXERCISE

SECTION 4-9, PAGE 154

```
insert into sec0409_employees
   (employee_id, first_name, last_name, dept_code, hire_date,
    credit_limit, phone_number, manager_id)
   values( 301, 'Ellen', 'Perkins', 'IT', #03-09-2009#, 20.00,
   null, 201)
```

SECTION 4-10, PAGE 157

```
insert into sec0410_data1
select * from sec0410_data2
```

SECTION 4-11, PAGE 159

```
update sec0411_employees
set credit_limit = 27
where credit_limit = 15
    or
      credit_limit is null;
```

SECTION 4-12, PAGE 160

```
delete from sec0412_employees
where employee_id between 202 and 205;
```

SECTION 4-14, PAGE 166

NOT A CODING EXERCISE

SECTION 4-15, PAGE 169

```
delete from sec0415_departments
where dept_code = 'sal';

delete from l_departments
where dept_code = 'sal';
```

CHAPTER 5 – THE DATA DICTIONARY AND OTHER ORACLE TOPICS

SECTION 5-2, PAGE 174

ORACLE SQL EXERCISE

SECTION 5-3, PAGE 178

ORACLE SQL EXERCISE

SECTION 5-5, PAGE 188

ORACLE SQL EXERCISE

SECTION 5-6, PAGE 190

ORACLE SQL EXERCISE

SECTION 5-12, PAGE 199

ORACLE SQL EXERCISE

SECTION 5-13, PAGE 201

ORACLE SQL EXERCISE

SECTION 5-14, PAGE 203

ORACLE SQL EXERCISE

SECTION 5-15, PAGE 206

ORACLE SQL EXERCISE

CHAPTER 6 – CREATING YOUR OWN TABLES

SECTION 6-7, PAGE 224

```
-- Create table
create table L_EMPLOYEES2
   (EMPLOYEE_ID  number,
    FIRST_NAME   text(10),
    LAST_NAME    text(10),
    DEPT_CODE    text(3),
    HIRE_DATE    date,
    CREDIT_LIMIT currency,
    PHONE_NUMBER text(4),
    MANAGER_ID   number);

-- Copy contents
insert into L_EMPLOYEES2
select * from l_employees;
```

SECTION 6-9, PAGE 228

```
-- create table
create table SEC0609_EMPLOYEES2
   (EMPLOYEE_ID  number,
    FIRST_NAME   text(10),
    LAST_NAME    text(10),
    DEPT_CODE    text(3),
    HIRE_DATE    date,
    CREDIT_LIMIT currency,
    PHONE_NUMBER text(4),
    MANAGER_ID   number);

-- copy contents
insert into SEC0609_EMPLOYEES2
select * from SEC0609_EMPLOYEES;

alter table SEC0609_EMPLOYEES2
add constraint pk_sec0609_employees
primary key (employee_id);
```

SECTION 6-11, PAGE 231

```
-- Create table
create table SEC0611_DEPARTMENTS2
   (DEPT_CODE          text(3),
    DEPARTMENT_NAME    text(30));

-- copy contents
insert into SEC0611_DEPARTMENTS2
select * from SEC0611_DEPARTMENTS;

-- add manager column
alter table SEC0611_DEPARTMENTS2
add column MANAGER text(10);

--add annual budget column
alter table SEC0611_DEPARTMENTS2
add column ANNUAL_BUDGET number;
```

SECTION 6-12, PAGE 232

```
-- create table
create table SEC0612_EMPLOYEES2
   (EMPLOYEE_ID  number,
    FIRST_NAME   text(10),
    LAST_NAME    text(10),
    DEPT_CODE    text(3),
    HIRE_DATE    date,
    CREDIT_LIMIT currency,
    PHONE_NUMBER text(4),
    MANAGER_ID   number);

-- copy contents
insert into SEC0612_EMPLOYEES2
select * from SEC0612_EMPLOYEES;

-- expand last name
alter table SEC0612_EMPLOYEES2
alter column LAST_NAME text(50);
```

SECTION 6-13, PAGE 233

```
-- create table
create table SEC0613_EMPLOYEES2
   (EMPLOYEE_ID  number,
    FIRST_NAME   text(10),
    LAST_NAME    text(10),
    DEPT_CODE    text(3),
    HIRE_DATE    date,
    CREDIT_LIMIT currency,
    PHONE_NUMBER text(4),
    MANAGER_ID   number);

-- copy contents
insert into SEC0613_EMPLOYEES2
select * from SEC0613_EMPLOYEES;

-- drop phone number column
alter table SEC0613_EMPLOYEES2
drop column PHONE_NUMBER;
```

SECTION 6-16, PAGE 240

```
--  store no-dupes data into temporary table
select distinct *
   into SEC0616_DUPLICATE_ROWS_TEMP
   from SEC0616_DUPLICATE_ROWS;

-- clear out the final destination table
drop table SEC0616_DUPLICATE_ROWS;

-- copy data from temporary to final
select *
   into SEC0616_DUPLICATE_ROWS
   from SEC0616_DUPLICATE_ROWS_TEMP;

-- erase temporary table
drop table SEC0616_DUPLICATE_ROWS_TEMP;
```

SECTION 6-17, PAGE 242

```
alter table SEC0617_DUPLICATE_ROWS
add column id counter;
```

CHAPTER 7 – FORMATS, SEQUENCES AND INDEXES

SECTION 7-2, PAGE 251

```
select employee_id,
       first_name,
       hire_date,
       format(hire_date, 'MMMM DD, YYYY')
       as formatted_date
from l_employees
order by employee_id;
```

SECTION 7-3, PAGE 254

```
-- create table
select *
into SEC0703_EMPLOYEES2
from SEC0703_EMPLOYEES;

-- Insert a single row
insert into SEC0703_EMPLOYEES2
(employee_id, first_name, hired_date)
values ('George', 'Washington', #July 4, 1776 10:00AM#));
```

SECTION 7-6, PAGE 258

ORACLE SQL EXERCISE

SECTION 7-7, PAGE 260

ORACLE SQL EXERCISE

SECTION 7-8, PAGE 262

```
alter table sec0708_departments
add column new_num counter;
```

SECTION 7-9, PAGE 264

```
-- create copy of table
select *
into SEC0709_DEPARTMENTS2
from SEC0709_DEPARTMENTS;

-- create index
create index ix_sec0709_dept_code
on SEC0709_DEPARTMENTS2 (DEPT_CODE);
```

SECTION 7-12, PAGE 269

NON-PROGRAMMING EXERCISE

SECTION 7-14, PAGE 274

NON-PROGRAMMING EXERCISE

SECTION 7-16, PAGE 276

ORACLE SQL EXERCISE

SECTION 7-17, PAGE 277

ORACLE SQL EXERCISE

CHAPTER 8 - DATA INTEGRITY – NOT COMPLETED

SECTION 8-2, PAGE 285

GUI BASED EXERCISE

SECTION 8-3, PAGE 286

GUI BASED EXERCISE

SECTION 8-4, PAGE 287

GUI BASED EXERCISE

SECTION 8-5, PAGE 289

GUI BASED EXERCISE

SECTION 8-8, PAGE 292

```
alter table sec0808_employees
add constraint fk_sec0808_employees_dept_code
foreign key (dept_code)
   references sec0808_departments ( dept_code);
```

SECTION 8-10, PAGE 295

```
insert into sec0809_employees
values( 211, 'Barak', 'Obama', 'pre', #Jan-01-2008#, 100, 555,
null);

update sec0809_employees
set dept_code = 'pre'
```

```
where employee_id = 210;
```

SECTION 8-11, PAGE 296

```
insert into sec0810_employees
values( 211, 'Barak', 'Obama', 'Exe', #Jan-01-2008#, 100, 555,
null);

update sec0810_employees
set dept_code = 'Exe'
where employee_id = 210;
```

SECTION 8-12, PAGE 297

```
delete from sec0811_departments
where dept_code = 'Exe';

update sec0811_departments
set dept_code = 'PRZ'
where dept_code = 'Exe';
```

SECTION 8-13, PAGE 299

```
update sec0812_employees
set dept_code = null
where dept_code = 'Shp';

delete from sec0812_departments
where dept_code = 'Shp';
```

SECTION 8-15, PAGE 302

GUI BASED EXERCISE

SECTION 8-17, PAGE 306

GUI BASED EXERCISE

SECTION 8-18, PAGE 308

Set the relationship in the GUI, then run the following commands:

```
delete from sec0818_departments
where dept_code = 'Shp';
```

SECTION 8-19, PAGE 311

Set the relationship in the GUI, then run the following commands:

```
update sec0819_departments
set dept_code = 'Abc'
where dept_code = 'Shp';
```

CHAPTER 9 – ROW FUNCTIONS

SECTION 9-2, PAGE 327

```
select *,
       first_name & ' ' & last_name as full_name,
       credit_limit + 10 as new_credit_limit
into SEC0902_EMPLOYEES
from l_employees;
```

SECTION 9-3, PAGE 329

Note: results from the solutions looks outdated.

```
select employee_id,
       first_name,
       last_name,
       credit_limit + 10 as new_credit_limit
from l_employees
where hire_date < #01-01-2000#
order by employee_id;
```

SECTION 9-4, PAGE 331

```
select employee_id,
       first_name,
       last_name,
       credit_limit + 10 as new_credit_limit
from l_employees
where credit_limit + 10 > 20
order by credit_limit + 10;
```

SECTION 9-5, PAGE 334

Step 1:

Create view

```
select employee_id,
       first_name,
       last_name,
       credit_limit + 10 as new_credit_limit
```

```
from l_employees;
```

Step 2:

Query new view

```
select *
from sec0905_employee_view
where new_credit_limit > 20
order by new_credit_limit;
```

SECTION 9-8, PAGE 339

```
select  n,
        5*n as [5*n]
from sec0908_test_numbers
order by n;


select  n,
        n/10 as [n/10]
from sec0908_test_numbers
order by n;


select  n,
        10/n as [10/n]
from sec0908_test_numbers
order by n;


select  n,
        10/n as [10/n]
from sec0908_test_numbers
where n <> 0
order by n;


select  n,
        2^n as [2^n]
from sec0908_test_numbers
order by n;
```

```
select  n,
        sqr(n) as [sqr(n)]
from sec0908_test_numbers
order by n;


select  n,
        sqr(n) as [sqr(n)]
from sec0908_test_numbers
where n >= 0
order by n;


select  n,
        n\3 as [n\3]
from sec0908_test_numbers
where n >= 0
order by n;


select  n,
        n mod 3 as [n mod 3]
from sec0908_test_numbers
order by n;
```

SECTION 9-9, PAGE 343

```
select 'first' & 'second',
       mid('abcdefghij', 3,4),
       len('abcdefg'),
       instr('abcdefg','cd'),
       instr('abcdefg','zz'),
       ucase('dog'),
       lcase('CAT'),
       trim('   bird   ');
```

SECTION 9-10, PAGE 345

```
select employee_id,
       last_name & ', ' & mid(first_name,1,1) & '.'
from l_employees
order by employee_id;
```

SECTION 9-11, PAGE 348

Create view **SEC0911_names_view1** to store the location of the last letter of the first name.

```
select full_name,
       instr(full_name, ', ') -1 as position
from sec0911_names;
```

Create another view **SEC0911_names_view2** based on the previous view to find the location of the last letter in the last name relative to the start of the first letter of the last name. This is relative distance not absolute.

```
SELECT a.full_name,
       instr( mid(a.full_name, a.position + 3, len(a.full_name)
                      - (a.position + 3) ),
              ' ') -1 AS [position]
FROM sec0911_names_view1 AS a;
```

Now, build the first name, last name and initial by using the relative locations of each part from inner joining the two views together.

```
select a.full_name,
       mid(a.full_name, 1, a.position) as first_name,
       mid(a.full_name, a.position+3, b.position)   as
last_name,
       mid(a.full_name, a.position+4+b.position,1 ) as
middle_name
from sec0911_names_view1 a,
     sec0911_names_view2 b
where a.full_name = b.full_name;
```

SECTION 9-12, PAGE 350

Note: In Access, the third column cannot have the same name, PHONE_NUMBER, as the table column for some reason or the query will not run properly even though I properly alias the table.

```
select a.phone_number,
       mid(a.phone_number, instr( a.phone_number, '(')+1,3) as
AREA_CODE,
       mid(a.phone_number, instr( a.phone_number, '(')+6,8) as
PHONENUMBER
```

```
from SEC0912_PHONE_NUMBERS AS a;
```

SECTION 9-13, PAGE 354

NOTE: The book incorrectly states the 'y' is the proper interval to add a year to a date. It is actually 'yyyy.'

```
select #07-mar-2011# as [DATE],
       #07-mar-2011# + 2 as PLUS_2D,
       #07-mar-2011# - 2 as MINUS_2D,
       DateAdd('m', 2, #07-mar-2011#) as PLUS_2M,
       DateAdd('yyyy', 2, #07-mar-2011#) as PLUS_2Y,
       #27-mar-2011# - #07-mar-2011# as DELTA;
```

SECTION 10-4, PAGE 372

Yes. Obviously, if you convert a number to a string then sort by it then it will sort like a string.

SECTION 10-5, PAGE 373

Oracle SQL SPECIFIC

SECTION 10-6, PAGE 374

NOT A CODING EXERCISE

SECTION 10-7, PAGE 376

GUI EXERCISE

SECTION 10-8, PAGE 379

Create a view **sec1008_view** for multiples of 7.

```
select n,
       n * 7 as multiple_of_7
from NUMBERS_0_TO_99999 a;
```

Now select those numbers in the proper range.

```
select n,
       multiple_of_7
from SEC1008_VIEW
where multiple_of_7 between 700 and 900
order by n;
```

SECTION 10-10, PAGE 383

I think the author wanted a solution with multiple dependent views. I think it is far simpler to try to replicate all the days of the current month and test to see if they are valid in the where clause.

```
select n
from NUMBERS_0_TO_99
where Month(DateSerial(Year(Date()), Month(Date()), n)) =
Month(Date())
order by n;
```

SECTION 10-12, PAGE 389

```
select now() - #Aug-04-1961# as days_since_birth,
       now() - #Jan-03-2004# as days_since_senate_inauguration,
       now() - #Jan-20-2009# as
days_since_president_1st_inauguration,
       now() - #Jan-20-2013# as
days_since_president_2nd_inauguration;
```

SECTION 10-13, PAGE 390

```
select #July-04-1776# + 100000 as celebration_day;
```

CHAPTER 11 – SUMMARIZING DATA

SECTION 11-3, PAGE 406

```
select min(num_1) as minimum,
       max(num_1) as maximum
from sec1103;
```

SECTION 11-4, PAGE 408

```
select min(num_1) as minimum,
       max(num_1) as maximum
from sec1103
where row_id < 8;
```

SECTION 11-5, PAGE 411

For minimum value.

```
select row_id,
       num_1
from sec1103
where num_1 = (select min(num_1)
               from sec1103);
```

For maximum value.

```
select row_id,
       num_1
from sec1103
where num_1 = (select max(num_1)
               from sec1103);
```

SECTION 11-6, PAGE 413

```
select count(*) as rows,
       count(num_1) as non_null_num_1,
       count(*) - count(num_1) as null_value
from sec1106;
```

SECTION 11-7, PAGE 415

```
select count(*) as rows,
       count(num_1) as non_null_num_1,
       count(*) - count(num_1) as null_value
from sec1106
where row_id = 1;
```

SECTION 11-8, PAGE 417

Create view **SEC1108_VIEW** using

```
select distinct num_1
from sec1103
```

Then run the following query that will ignore NULLs

```
select count(num_1) as rows
from SEC1108_VIEW;
```

SECTION 11-9, PAGE 420

Create view **SEC1103_VIEW** using

```
select distinct num_1,
       num_2
from SEC1103;
```

Then run the following query

```
select count(*)
from SEC1103_VIEW;
```

SECTION 11-10, PAGE 421

```
select sum(num_1) as sum,
       avg(num_1) as avg
from SEC1103;
```

SECTION 11-11, PAGE 427

Adding rows first

```
select sum(num_1 + num_2)
from SEC1103;
```

Adding column first

```
select sum(num_1) + sum(num_2)
from SEC1103;
```

Redo row wise addition with NZ to convert NULLS to 0.

```
select sum(nz(num_1,0) + nz(num_2,0))
from SEC1103;
```

CHAPTER 12 – CONTROLLING THE LEVEL OF SUMMARIZATION

SECTION 12-2, PAGE 441

```
select col_1,
       sum(col_3)
from SEC1202
group by col_1;
```

SECTION 12-4, PAGE 447

Note: The results from the query differ from the solutions provided by author.

```
select col_1,
       col_2,
       sum(col_3)
from SEC1202
group by col_1,col_2;
```

SECTION 12-7, PAGE 454

Not really a problem posed but more of a comment by author on the state of SQL databases.

SECTION 12-11, PAGE 462

```
select col_1,
       sum(col_2)
from SEC1211
group by col_1
having sum(col_2) > 20
order by col_1;
```

CHAPTER 13 – INNER JOINS

SECTION 13-3, PAGE 482

NON-PROGRAMMING EXERCISE

SECTION 13-4, PAGE 484

NON-PROGRAMMING EXERCISE

SECTION 13-5, PAGE 486

NON-PROGRAMMING EXERCISE

SECTION 13-6, PAGE 489

NON-PROGRAMMING EXERCISE

SECTION 13-8, PAGE 492

NON-PROGRAMMING EXERCISE

SECTION 13-9, PAGE 495

```
select a.adjective,
       b.animal
from SEC1309_TABLE1 a,
     SEC1309_TABLE2 b
where left(a.adjective,1) = left(b.animal,1)
order by a.adjective;
```

SECTION 13-10, PAGE 497

```
select a.M1,
```

```
       a.M2,
       a.M3,
       a.Adjective,
       b.M1,
       b.M2,
       b.M3,
       b.Animal
from SEC1310_TABLE1 a,
     SEC1310_TABLE2 b
where (a.M1 = b.M1)
    and
      (a.M2 = b.M2)
    and
      (a.M3 = b.M3);
```

SECTION 13-11, PAGE 499

```
select a.Word,
       b.Beginning_Letter,
       b.Ending_Letter,
       b.Dictionary_Volume
from SEC1311_WORDS a,
     SEC1311_DICTIONARY b
where left(a.Word,1) between b.Beginning_Letter and
b.Ending_Letter
order by a.Word;
```

SECTION 13-12, PAGE 500

```
select a.Letter
from alphabet a
where a.Letter > "S"
order by a.Letter;
```

SECTION 13-13, PAGE 501

```
select a.word,
       b.word
from SEC1313_WORDS1 a,
   SEC1313_WORDS2 b
where right(a.word,1) = right(b.word,1)
order by a.word;
```

SECTION 13-14, PAGE 504

It's important to include the brackets around the ON statement or else the parser will fail!

```
select a.student_name,
       a.test_score,
       b.letter_grade
from sec1311_student_scores a
     inner join sec1311_grade_ranges b
     on (a.test_score between
        b.beginning_score and b.ending_score)
order by a.student_name;
```

SECTION 13-15, PAGE 507

```
-- old way
select a.description as FOOD,
       b.supplier_name as SUPPLIER_NAME
from l_foods a,
     l_suppliers b
where a.supplier_id = b.supplier_id
order by a.description;

-- new way
select a.description as FOOD,
       b.supplier_name as SUPPLIER_NAME
from l_foods a
     inner join l_suppliers b
        on a.supplier_id = b.supplier_id
order by a.description;
```

SECTION 13-16, PAGE 510

NOTE: I THINK THE SAMPLE OUTPUT PROVIDED BY THE AUTHOR IS WRONG: HE IS MISSING HIS OWN POINT. IN THIS SECTION, HE STATES A SINGLE SQL QUERY CAN BE BROKEN UP AS TWO QUERIES WHERE ONE QUERY CREATES A LARGER TABLE WITH ALL THE COLUMNS AND THE OTHER QUERY TO SELECT FROM THAT DATA. IN HIS SOLUTION, HIS FIRST STEP HAS ALREADY REDUCED THE DATA BY INCLUDING A WHERE CLAUSE. BY IMPLICATION -- AS HE DOES NOT PROVIDE ANY ACTUAL SQL CODE -- THE SECOND QUERY IS JUST A SIMPLE "SELECT * FROM TABLE" WITH NO WHERE CLAUSE AT ALL. THERE IS NO REDUCTION OF DATA, SO THE AUTHOR'S SOLUTION IS WRONG.

Step1:

Create a view in Access with name **sec1316_view** using the following code.

```
select a.description,
       a.supplier_id as a_id,
       b.supplier_name,
       b.supplier_id as b_id
from l_foods a,
     l_suppliers b;
```

Step 2:

Run query.

```
select description,
       supplier_name
from sec1316_view
where a_id = b_id
order by description;
```

SECTION 13-17, PAGE 511

NOTE: I think the solution's numbers are a bit off. Most of the numbers match exactly while a few are off. It's likely that the table data is out of sync when the original author's solution was written.

```
select b.description as FOOD,
       sum(a.quantity) as TOTAL_ORDERS
from l_lunch_items a,
     l_foods b
where a.product_code = b.product_code
group by b.description;
```

SECTION 13-19, PAGE 515

```
select a.*,
       b.*,
       c.*,
       d.*
from l_employees a,
     l_lunches b,
```

```
     l_lunch_items c,
     l_foods d
where a.employee_id = b.employee_id
  and b.lunch_id = c.lunch_id
  and c.supplier_id = d.supplier_id
  and c.product_code = d.product_code;
```

CHAPTER 14 - OUTER JOINS

SECTION 14-3, PAGE 524

```
select a.department_name,
       b.first_name,
       b.last_name
from l_departments a
     left join l_employees b
     on a.dept_code = b.dept_code;
```

The difference between inner join and a left outer join is the additional row for "PERSONNEL" dept.

SECTION 14-4, PAGE 526

```
select a.department_name,
       b.first_name,
       b.last_name
from l_employees b
     right outer join l_departments a
     on a.dept_code = b.dept_code;
```

The difference between inner join and a right outer join is the additional row for "PERSONNEL" dept.

SECTION 14-5, PAGE 528

```
select a.department_name,
       b.first_name,
       b.last_name
from l_departments a
     left outer join sec1405_employees b
     on a.dept_code = b.dept_code
union
select a.department_name,
       b.first_name,
       b.last_name
from l_departments a
     right outer join sec1405_employees b
     on a.dept_code = b.dept_code;
```

SECTION 14-7, PAGE 533

```
select *
   from sec1407_departments
union
select *
   from l_departments
order by dept_code;
```

SECTION 14-8, PAGE 536

```
select a.description as food_item,
       nz(count(b.quantity),0) as number_of_orders
from l_foods a
     left outer join l_lunch_items b
     on (a.supplier_id = b.supplier_id) and
        (a.product_code = b.product_code)
group by a.description
order by a.description;
```

Line two can be written this way too.

```
      count(b.quantity) * 1 as number_of_orders
```

SECTION 14-12, PAGE 546

Note: The solution provided by the author differs from the database information also provided by the author.

```
select a.dept_code ,
          a.department_name ,
          b.dept_code,
          b.department_name
from l_departments a
         left outer join sec1412_departments b
         on a.dept_code = b.dept_code and
               a.department_name = b.department_name
where b.dept_code is null or
           b.department_name is null
union
select a.dept_code ,
          a.department_name ,
          b.dept_code,
```

```
          b.department_name
from sec1412_departments a
         left outer join l_departments b
         on a.dept_code = b.dept_code and
               a.department_name = b.department_name
where b.dept_code is null or
           b.department_name is null
order by a.dept_code, a.department_name;
```

CHAPTER 15 – UNION AND UNION ALL

SECTION 15-1, PAGE 559

1. The maximum size of the union of these tables is 150,000. The maximum size of the inner join of these tables is 50,000,000,000.
2. Inner join using select:

```
select *
   from twos
union
select *
from threes b
order by number_2;
```

SECTION 15-2, PAGE 561

```
select *
   from twos
union all
select *
from threes b
order by number_2;
```

The difference between **union** and **union all** is the addition of several duplicate rows. They are highlighted in blue below.

NUMBER_2	WORD_2
	Null
	Null
2	Two
3	Three
4	Four
6	Six
6	Six
8	Eight
9	Nine
10	Ten
12	Twelve

12	Twelve
14	Fourteen
15	Fifteen
16	Sixteen
18	Eighteen
18	Eighteen
20	Twenty

SECTION 15-3, PAGE 563

1. The number of columns must match
2. See item (3) below.
3. Goal 2:

```
select a.first_name,
           a.last_name,
           count(b.lunch_id) as number_of_lunches
from l_employees a
         inner join l_lunches b
         on a.employee_id = b.employee_id
group by a.first_name,
                a.last_name;
union all
select 'Carol',
           'Rose',
           0
from dual
union all
select 'Paula',
           'Jacobs',
           0
from dual;
```

SECTION 15-4, PAGE 567

In Access all four methods of specifying sort order works.

```
order by a.last_name;
order by last_name;
order by ln;
order by 2;
```

SECTION 15-5, PAGE 570

First, in Access, create a view called **SEC1505_EMPLOYEES_VIEW** with the following SQL code:

```
select *
from l_employees
union
select 301, 'Gail', 'Jones','Sal',#02-15-2011#,25,null, 202
from l_employees;
```

Then copy all the data into a new table using the following code.

```
select *
into SEC1505_EMPLOYEES
from SEC1505_EMPLOYEES_VIEW;
```

SECTION 15-6, PAGE 573

Access does not support data dictionaries. However, it is possible to save the second query into a new table. By examining this table using "Design View" we notice that the numeric column has precision of 7 while the text column now has a length of 255. It seems implicit conversion for numbers (max precision of two **unioned** columns) works differently than text (defaults to 255) in Microsoft Access.

SECTION 15-7, PAGE 575

```
select format(date_1),
       format(date_1),
       format(date_1)
from sec1507_first
union
select format(number_2),
       format(word_2),
       format(date_2)
from sec1507_second;
```

SECTION 15-8, PAGE 577

The code doesn't actually run properly. I think the database provided is out date as compared to that used in the book.

```
select number_1,
       word_1,
       date_1
from sec1508_more_columns
union
select number_2,
       word_2,
       null
from twos;
```

SECTION 15-9, PAGE 580

Step 1:

First determine number of rows in each table.

```
select count(*) from sec1509a_lunches;
select count(*) from sec1509b_lunches;
select count(*) from l_lunches;
```

All tables contain 16 rows. We need to dig deeper.

Step 2:

Create a view named **SEC1509_UNION_VIEWA** with the following code.

```
select * from l_lunches
union
select * from sec1509a_lunches;
```

Create a view named **SEC1509_UNION_VIEWB** with the following code.

```
select * from l_lunches
union
select * from sec1509b_lunches;
```

Step 3:

Count the number of rows for each view using the following SQL code.

```
select count(*)
from sec1509_union_viewa;

select count(*)
from sec1509_union_viewb;
```

The first and second count on the views returns 17 and 16 rows, respectively. As the source itself has 16 rows, **sec1509b_lunches** must be identical to **l_lunches** table.

SECTION 15-10, PAGE 583

```
select number_2,
       word_2,
       'from Twos table'
from twos
union
select number_3,
       word_3,
       'from Threes table'
from threes;
```

SECTION 15-11, PAGE 585

```
select *,
       '' as MESSAGE
from l_employees
where year(hire_date) >= 2000

union
select *,
       'Old Timer' as MESSAGE
from l_employees
where year(hire_date) < 2000

order by employee_id;
```

SECTION 15-12, PAGE 586

The solution provided by the author includes too many columns and is sorted by **EMPLOYEE_ID**. This solution is based on the question actually asked by the book.

```
select first_name,
       last_name,
       '' as OLD_TIMERS,
       hire_date as NEWER_HIRES
from l_employees
where year(hire_date) >= 2000

union

select first_name,
       last_name,
       hire_date as OLD_TIMERS,
       ''
from l_employees
where year(hire_date) < 2000

order by last_name, first_name;
```

SECTION 15-13, PAGE 588

```
select digit * 2
   from NUMBERS_0_TO_9
   where digit mod 2 = 0

union

select digit * 3
   from NUMBERS_0_TO_9
   where digit mod 2 = 1
```

SECTION 15-15, PAGE 592

```
select a.*, b.*
from twos a,
     threes b
where a.number_2 = b.number_3 and
      a.word_2 = b.word_3;
```

SECTION 15-16, PAGE 596

```
select a.*
from twos a
     left outer join threes b
     on a.number_2 = b.number_3 and
        a.word_2 = b.word_3
where b.word_3 is null;
```

CHAPTER 16 – CROSS JOINS, SELF JOINS, AND CROSSTAB QUERIES

SECTION 16-1, PAGE 601

THE ANSWER IS CONCEPTUALLY SIMILAR TO PROBLEM 16-3

SECTION 16-3, PAGE 604

2	(null)
2	3
2	6
2	9
2	12
4	(null)
4	3
4	6
4	9
4	12
6	(null)
6	3
6	6
6	9
6	12
8	(null)
8	3
8	6
8	9
8	12
10	(null)
10	3
10	6
10	9
10	12
12	(null)
12	3
12	6
12	9
12	12

SECTION 16-5, PAGE 608

If the *where* clause is missing a condition to join the two tables, then it causes an additional 4 times more results in the output.

SECTION 16-11, PAGE 621

```
select a.sequence_id,
       a.prime_number - b.prime_number
from sec1611_prime_numbers a,
     sec1611_prime_numbers b
where a.sequence_id = b.sequence_id + 1
order by a.sequence_id;
```

CHAPTER 18 – IF-THEN-ELSE, PARAMETER QUERIES, AND SUBQUERIES

SECTION 18-1, PAGE 679

ORACLE EXERCISE

SECTION 18-2, PAGE 682

```
select employee_id,
       first_name,
       last_name,
       iif ( dept_code = 'exe', 'EXECUTIVE',
       iif ( dept_code = 'sal', 'SALES',
       iif ( dept_code = 'shp', 'SHIPPING',
       iif ( dept_code = 'act', 'ACCOUNTING','MARKETING')
       )  )  ) as Department
from l_employees;
```

SECTION 18-3, PAGE 685

```
select employee_id,
       first_name,
       last_name,
       hire_date,
       iif(year(hire_date) <= 2005, 'Old Guard', 'Young Turk')
as message
from l_employees;
```

SECTION 18-4, PAGE 687

```
select employee_id,
       first_name,
       last_name,
       hire_date,
       iif(year(hire_date) <= 2005, hire_date, '') as OLD_GUARD,
       iif(year(hire_date) <= 2005, '', hire_date) as YOUNG_TURK
from l_employees;
```

SECTION 18-5, PAGE 689

```
select employee_id,
       first_name,
       last_name,
       hire_date,
       credit_limit,
       credit_limit * iif(year(hire_date) < 2000, 2, 1.5) as
new_credit_limit
from l_employees;
```

SECTION 18-6, PAGE 693

ORACLE EXERCISE

SECTION 18-7, PAGE 697

ORACLE EXERCISE

SECTION 18-8, PAGE 698

ORACLE EXERCISE

SECTION 18-9, PAGE 699

```
select department_name
from l_departments
where dept_code = code;
```

SECTION 18-10, PAGE 700

```
select b.department_name,
       a.employee_id,
```

```
       a.first_name,
       a.last_name
from l_employees a,
     l_departments b
where a.dept_code = b.dept_code and
      a.dept_code = [pick a department];
```

SECTION 18-12, PAGE 703

```
select *
from l_employees
where credit_limit > ( select avg(credit_limit)
                       from l_employees);
```

SECTION 18-13, PAGE 705

Subquery

```
select employee_id
from l_employees;
```

Whole query

```
select *
from l_lunches
where employee_id in ( select employee_id
                       from l_employees);
```

SECTION 18-14, PAGE 707

```
select *
from l_employees
where credit_limit > ( select avg(credit_limit)
                       from l_employees);
```