EXERCISE SOLUTIONS

The solutions follow these guidelines (though they are not necessary to get the correct result):

- Joined tables always have correlation names (e.g. p or e) and all columns are qualified with the correlation name (e.g. p.project_name)
- All columns with calculations and functions are given an alias name (e.g. SELECT MAX(salary) AS max_salary …)
- Missing values are marked as NULL (your product might do otherwise)
- If there is no ORDER BY clause, the order in your product may be different

6.1 Fundamentals

1.

```
SELECT *
FROM department
```

```
DEPTNO DEPTNAME        CODE
------ --------------- ------------
     1 IT              asa_123456
     2 Economy         s'dfg*234
     3 Research        a_ss*8888
     4 Marketing       a%
```

2.

```
SELECT deptno, deptname
FROM department
```

```
DEPTNO DEPTNAME
------ ---------------
     4 Marketing
     2 Economy
     1 IT
     3 Research
```

3.

```
SELECT DISTINCT location
FROM project
```

```
LOCATION
---------------
HELSINKI
SINGAPORE
LONDON
NULL
```

4.

```
SELECT fname, lname, city,
       salary AS monthly_salary
FROM employee
```

FNAME	LNAME	CITY	MONTHLY_SALARY
Peter	Stream	LONDON	2800
Mike	Wood	LONDON	3100
Rachel	Brooke	HELSINKI	3100
Leon	Lake	LONDON	2800
Peter	Taylor	HELSINKI	2960
Laura	Brown	SYDNEY	2650
Lilian	River	SYDNEY	2800

5.

```
SELECT fname, lname, city,
salary AS monthly_salary
FROM employee
ORDER BY city, lname
```

FNAME	LNAME	CITY	MONTHLY_SALARY
Rachel	Brooke	HELSINKI	3100
Peter	Taylor	HELSINKI	2960
Leon	Lake	LONDON	2800
Peter	Stream	LONDON	2800
Mike	Wood	LONDON	3100
Laura	Brown	SYDNEY	2650
Lilian	River	SYDNEY	2800

6.

```
SELECT 'project' AS text1, project_name,
       'is located in' AS text2,
       location
FROM project
```

text1	project_name	text2	location
project	BOOKKEEPING	is located in	LONDON
project	BILLING	is located in	HELSINKI
project	WAREHOUSING	is located in	HELSINKI
project	ACCOUNTING	is located in	LONDON
project	CUSTOMERS	is located in	SINGAPORE
project	STATISTICS	is located in	NULL

6.2 Parenthesis

1.

```
SELECT lname, fname, education
FROM employee
WHERE education = 'Ba'
```

```
LNAME      FNAME      EDUCATION
---------- ---------- --------
Stream     Peter      Ba
Brown      Laura      Ba
```

2.

```
SELECT fname, lname, salary
FROM employee
WHERE salary < 2960
```

```
FNAME      LNAME          SALARY
---------- ---------- ----------
Peter      Stream           2800
Leon       Lake             2800
Laura      Brown            2650
Lilian     River            2800
```

3.

```
SELECT fname, lname, salary
FROM employee
WHERE salary <= 2960
ORDER BY salary DESC
```

```
FNAME      LNAME          SALARY
---------- ---------- ----------
Peter      Taylor           2960
Peter      Stream           2800
Leon       Lake             2800
Lilian     River            2800
Laura      Brown            2650
```

4.

```
SELECT fname, lname, city, deptno
FROM employee
WHERE deptno = 3
AND city = 'LONDON'
ORDER BY lname, fname
```

```
FNAME      LNAME      CITY           DEPTNO
---------- ---------- ---------- ----------
Leon       Lake       LONDON              3
Peter      Stream     LONDON              3
```

5.

```
SELECT empno, lname, fname, city, salary
 FROM employee
 WHERE city = 'LONDON'
 OR salary = 2800
```

```
EMPNO LNAME     FNAME      CITY           SALARY
----  --------- ---------- ---------- ----------
2134  Stream    Peter      LONDON           2800
2234  Wood      Mike       LONDON           3100
2345  Lake      Leon       LONDON           2800
3547  River     Lilian     SYDNEY           2800
```

6.3 String Searches

1.

```
SELECT *
FROM employee
WHERE salary <> 2800
```

```
EMPNO FNAME  LNAME  CITY     EDUCATION SALARY TAX_RATE START_DATE DEPTNO
----  -----  -----  -------- --------  ------ -------- ---------- -----
2234  Mike   Wood   LONDON   PhD         3100       33 15.10.2009     1
2245  Rachel Brooke HELSINKI MA          3100       31 24.09.2014     4
2884  Peter  Taylor HELSINKI MA          2960       31 12.05.2009  NULL
3546  Laura  Brown  SYDNEY   Ba          2650       22 15.09.2017     1
```

```
The WHERE clause can also be formatted like this:
           WHERE NOT (salary = 2800)
           WHERE NOT salary = 2800
           WHERE salary != 2800
           WHERE salary NOT IN (2800)
```

2.

```
SELECT lname, fname, city, deptno, tax_rate
  FROM employee
  WHERE deptno = 3
  AND (city = 'SYDNEY' OR tax_rate = 22)
```

```
LNAME      FNAME      CITY           DEPTNO   TAX_RATE
---------- ---------- ---------- ---------- ----------
Stream     Peter      LONDON              3         22
River      Lilian     SYDNEY              3         37
```

3.

```
SELECT lname, fname, education, salary
FROM employee
WHERE (education = 'Ba' OR education = 'MA')
AND (salary = 3100 OR salary = 2800)
```

```
LNAME      FNAME      EDUCATION    SALARY
---------- ---------- -------- ----------
Stream     Peter      Ba             2800
Brooke     Rachel     MA             3100
```

```
Education condition can also be done like this:
          WHERE education IN ('Ba', 'MA')
Likewise the salary condition.
```

4.

```
SELECT *
FROM project
WHERE project_name LIKE 'B%'
```

```
projno  project_name  priority   location
------- ------------- ---------- --------
P1      BOOKKEEPING   2          LONDON
P2      BILLING       1          HELSINKI
```

5.

```
SELECT empno, lname, fname, city, salary
FROM employee
WHERE city LIKE '%I%'
```

```
empno      lname      fname      city       salary
---------  --------   ---------  ---------  --------
2245       Brooke     Rachel     HELSINKI   3100.00
2884       Taylor     Peter      HELSINKI   2960.00
```

6.

```
SELECT empno, fname, lname
FROM employee
WHERE NOT lname LIKE '_a%'   -- or WHERE lname NOT LIKE '_a%'
```

```
empno      fname      lname
---------  ---------- -------
2245       Rachel     Brooke
2234       Mike       Wood
2134       Peter      Stream
3546       Laura      Brown
3547       Lilian     River
```

7.

```
SELECT empno, lname, fname, tax_rate
FROM employee
WHERE tax_rate BETWEEN 22 AND 31
ORDER BY tax_rate
```

```
EMPNO LNAME      FNAME        TAX_RATE
----  ---------- ---------- ----------
2134  Stream     Peter              22
3546  Brown      Laura              22
2345  Lake       Leon             24,5
2245  Brooke     Rachel             31
2884  Taylor     Peter              31
```

8.

```
SELECT empno, lname, fname, tax_rate
FROM employee
WHERE tax_rate IN (31, 24.5, 37)
ORDER BY tax_rate
```

```
EMPNO LNAME      FNAME      TAX_RATE
----  ---------- ---------- ----------
2345  Lake       Leon             24,5
2245  Brooke     Rachel             31
2884  Taylor     Peter              31
3547  River      Lilian             37
```

9.

```
SELECT empno, lname, fname, education
FROM employee
WHERE education IS NULL
```

```
EMPNO LNAME      FNAME      EDUCATION
----  ---------- ---------- --------
2345  Lake       Leon       NULL
```

10.

```
SELECT empno, lname, fname, education
FROM employee
WHERE education <> 'MA'      -- or  WHERE NOT education = 'MA'
OR education IS NULL
```

```
EMPNO LNAME      FNAME      EDUCATION
----- ---------- ---------- --------
2134  Stream     Peter      Ba
2234  Wood       Mike       PhD
2345  Lake       Leon       NULL
3546  Brown      Laura      Ba
3547  River      Lilian     DIP
```

6.4. Functions

1.

```
SELECT MAX(priority) AS maximum, MIN(priority) AS minimum
FROM project
```

```
   MAXIMUM    MINIMUM
---------- ----------
         3          1
```

2.

```
SELECT COUNT(*) AS cnt
FROM project
```

```
  CNT
-----
    6
```

3.

```
SELECT AVG(tax_rate) AS avg_tax
FROM employee
```

```
   AVG_TAX
----------
28,6428571
```

4.

```
SELECT SUM(salary) AS tot_salary
FROM employee
```

```
TOT_SALARY
----------
     20210
```

5.

```
SELECT COUNT(DISTINCT education) AS cnt
FROM employee
```

```
       CNT
----------
         4
```

6.

SQL Server, Snowflake, MySQL, Hive, PostgreSQL:

```
SELECT SUBSTRING(lname,1 ,3) AS first3
FROM employee
```

```
FIRST3
------
Str
Woo
Bro
Lak
Tay
Bro
Riv
```

or

```
SELECT LEFT (lname, 3) AS first3
FROM employee
```

Oracle, DB2 (also Snowflake, MySQL, PostgreSQL, Hive):

```
SELECT SUBSTR(lname, 1, 3) AS first3
FROM employee
```

Since there is no ORDER BY the row order is random.

7.

Oracle, DB2, Snowflake, PostgreSQL

```
SELECT lname || ', ' || fname AS name
FROM employee
```

```
NAME
-------------
Taylor, Peter
Brooke, Rachel
Wood, Mike
Lake, Leon
Stream, Peter
Brown, Laura
River, Lilian
```

SQL Server:

```
SELECT lname + ', ' + fname AS name
FROM employee
```

MySQL, Hive:

```
SELECT CONCAT(lname, ', ', fname) AS name
FROM employee
```

8.

SQL Server:

```
SELECT lname, fname,
      UPPER(SUBSTRING(lname, 1, 2) + SUBSTRING(fname, 1, 2)) AS user_id
FROM employee
```

```
lname      fname      user_id
---------  --------   -------
Taylor     Peter      TAPE
Brooke     Rachel     BRRA
Wood       Mike       WOMI
Lake       Leon       LALE
Stream     Peter      STPE
Brown      Laura      BRLA
River      Lilian     RILI
```

Oracle, DB2:

```
SELECT lname, fname,
       UPPER(SUBSTR(lname, 1, 2) || SUBSTR(fname, 1, 2)) AS user_id
FROM employee
```

PostgreSQL, Snowflake:

```
SELECT lname, fname,
       UPPER(SUBSTRING(lname, 1, 2) || SUBSTRING(fname, 1, 2)) AS user_id
FROM employee
```

MySQL, Hive:

```
SELECT lname, fname,
  UPPER(CONCAT(SUBSTRING(lname, 1, 2), SUBSTRING(fname, 1, 2))) AS user_id
FROM employee
```

6.5 Dates

1.

SQL Server, PostgreSQL, Snowflake, Hive:

```
SELECT lname, fname, start_date
FROM employee
WHERE start_date > '2017-09-14'
```

```
lname      fname      start_date
---------  ---------  -----------------------
Stream     Peter      2020-03-02 00:00:00.000
Lake       Leon       2018-01-01 00:00:00.000
Brown      Laura      2017-09-15 00:00:00.000
```

Oracle, (PostgreSQL, Snowflake):

```
SELECT empno, lname, fname, start_date
FROM employee
WHERE start_date > TO_DATE('2017-09-14', 'YYYY-MM-DD')
```

or

```
SELECT empno, lname, fname, start_date
FROM employee
WHERE start_date > DATE '2017-09-14'
```

DB2:

```
SELECT empno, lname, fname, start_date
FROM employee
WHERE VARCHAR_FORMAT (start_date, 'YYYY-MM-DD') > '2017-09-14'
```

2.

Day is 2025-01-06

SQL Server:

```
SELECT YEAR(GETDATE()) AS year,
       MONTH(GETDATE()) AS month,
       DAY (GETDATE()) AS day
```

```
year       month      day
---------  ---------- ---
2025       1          6
```

Oracle:

```
SELECT EXTRACT (YEAR FROM SYSDATE) AS year,
       EXTRACT (MONTH FROM SYSDATE) AS mon,
       EXTRACT (DAY FROM SYSDATE) AS day;
```

PostrgreSQL:

```
SELECT EXTRACT (YEAR FROM CURRENT_DATE) as year,
       EXTRACT (MONTH FROM CURRENT_DATE) AS mon,
       EXTRACT (DAY FROM CURRENT_DATE) AS day
```

```
MySQL, Snowflake, Hive:

SELECT YEAR(CURRENT_DATE) AS year,
       MONTH(CURRENT_DATE) AS month,
       DAY (CURRENT_DATE) AS day

DB2:

SELECT YEAR(CURRENT_DATE) AS year,
       MONTH(CURRENT_DATE) AS month,
       DAY (CURRENT_DATE) AS day
FROM sysibm.sysdummy1
```

6.6 Grouping

```
1.

SELECT city, SUM(salary) AS total
FROM employee
GROUP BY city

city       total
---------  --------
HELSINKI   6060.00
LONDON     8700.00
SYDNEY     5450.00

2.

SELECT location, COUNT(*) AS cnt
FROM project
GROUP BY location

LOCATION               CNT
---------------  ----------
HELSINKI                 2
SINGAPORE                1
LONDON                   2
NULL                     1

The order of rows may differ in your product.


3.

SELECT location, COUNT(*) AS cnt
FROM project
GROUP BY location
HAVING COUNT(*) >= 2

LOCATION               CNT
---------------  ----------
HELSINKI                 2
LONDON                   2
```

6.7 Calculation with columns

1.

```
SELECT empno, lname, fname, salary, tax_rate,
       salary*tax_rate/100 AS tax
FROM employee
ORDER BY tax DESC
```

```
EMPNO LNAME      FNAME          SALARY   TAX_RATE        TAX
----  ---------- ---------- ---------- ---------- ----------
3547  River      Lilian           2800         37       1036
2234  Wood       Mike             3100         33       1023
2245  Brooke     Rachel           3100         31        961
2884  Taylor     Peter            2960         31      917,6
2345  Lake       Leon             2800       24,5        686
2134  Stream     Peter            2800         22        616
3546  Brown      Laura            2650         22        583
```

2.

```
SELECT MAX(salary) AS max_salary, MIN(salary) AS min_salary,
       MAX(salary) - MIN(salary) AS diff,
      (MAX(salary) - MIN(salary)) * 100 / MIN(salary) AS pros
FROM employee
```

```
max_salary min_salary diff       pros
---------- ---------- ---------- -------------
3100.00    2650.00    450.00     16.9811320754
```

3.

```
SELECT lname, salary, salary * 1.1 AS raised_sal
FROM employee
WHERE salary * 1.1 > 3000
```

```
LNAME          SALARY RAISED_SAL
---------- ---------- ----------
Stream           2800       3080
Wood             3100       3410
Brooke           3100       3410
Lake             2800       3080
Taylor           2960       3256
River            2800       3080
```

7.1 Join syntax

1.

```
SELECT p.project_name, pe.hours_act
FROM project p
  JOIN proj_emp pe
    ON p.projno = pe.projno
WHERE pe.empno = '2245'
```

```
PROJECT_NAME     HOURS_ACT
-------------- ----------
BOOKKEEPING            200
BILLING                400
WAREHOUSING            900
ACCOUNTING             200
```

```
Traditional syntax:
SELECT p.project_name, pe.hours_act
FROM project p, proj_emp pe
WHERE p.projno = pe.projno
AND pe.empno = '2245'
```

2.

```
SELECT p.project_name, p.projno, pe.empno,
        pe.hours_act, pe.hours_est
FROM project p
  JOIN proj_emp pe
    ON p.projno = pe.projno
ORDER BY p.project_name
```

```
PROJECT_NAME    PROJNO EMPNO HOURS_ACT  HOURS_EST
--------------- ------ ----- ---------- -----------
ACCOUNTING      P4     2245      200        200
ACCOUNTING      P4     2234      300        400
ACCOUNTING      P4     2884      400        600
BILLING         P2     2245      400        500
BILLING         P2     2134      300        NULL
BOOKKEEPING     P1     2345      100        100
BOOKKEEPING     P1     2234      200        NULL
BOOKKEEPING     P1     2884      100        200
BOOKKEEPING     P1     3546      400        500
BOOKKEEPING     P1     2245      200        300
BOOKKEEPING     P1     2134      300        300
BOOKKEEPING     P1     3547      300        200
WAREHOUSING     P3     2245      900        100
```

```
Traditional syntax:
SELECT p.project_name, p.projno, pe.empno,
        pe.hours_act, pe.hours_est
FROM project p, proj_emp pe
WHERE p.projno = pe.projno
ORDER BY p.project_name
```

3.

```
SELECT d.deptname, d.deptno, e.lname, e.fname,
       e.city, e.salary
FROM department d
  JOIN employee e
    ON d.deptno = e.deptno
WHERE e.salary <= 2800
ORDER BY d.deptname, e.lname, e.fname
```

```
DEPTNAME      DEPTNO   LNAME     FNAME     CITY       SALARY
------------- -------- --------  --------- ---------- ----------
IT                   1 Brown     Laura      SYDNEY     2,650
Research             3 Lake      Leon       LONDON     2,800
Research             3 River     Lilian     SYDNEY     2,800
Research             3 Stream    Peter      LONDON     2,800
```

```
Traditional syntax:
SELECT d.deptname, d.deptno, e.lname, e.fname,
       e.city, e.salary
FROM department d, employee e
WHERE d.deptno = e.deptno
AND   e.salary <= 2800
ORDER BY d.deptname, e.lname, e.fname
```

4.

```
SELECT p.project_name, e.lname, pe.hours_act
FROM employee e
   JOIN proj_emp pe
     ON e.empno = pe.empno
   JOIN project p
     ON p.projno = pe.projno
ORDER BY p.project_name, e.lname
```

```
PROJECT_NAME    LNAME      HOURS_ACT
--------------- ---------- ----------
ACCOUNTING      Brooke            200
ACCOUNTING      Taylor            400
ACCOUNTING      Wood              300
BILLING         Brooke            400
BILLING         Stream            300
BOOKKEEPING     Brooke            200
BOOKKEEPING     Brown             400
BOOKKEEPING     Lake              100
BOOKKEEPING     River             300
BOOKKEEPING     Stream            300
BOOKKEEPING     Taylor            100
BOOKKEEPING     Wood              200
WAREHOUSING     Brooke            900
```

```
Traditional syntax:
SELECT p.project_name, e.lname, pe.hours_act
FROM employee e, proj_emp pe, project p
WHERE e.empno = pe.empno
AND p.projno = pe.projno
ORDER BY p.project_name, e.lname
```

7.4 Outer join

1.

```
SELECT d.deptname, d.deptno,
       e.fname, e.lname
FROM department d
   LEFT JOIN employee e
     ON d.deptno = e.deptno
ORDER BY d.deptno, e.lname
```

```
deptname   deptno  fname         lname
---------- ------- ------------- ------
IT         1       Laura         Brown
IT         1       Mike          Wood
Economy    2       NULL          NULL
Research   3       Leon          Lake
Research   3       Lilian        River
Research   3       Peter         Stream
Marketing  4       Rachel        Brooke
```

2.

```
SELECT  e.lname, e.fname, d.deptname, d.deptno
FROM department d
   RIGHT JOIN employee e
     ON d.deptno = e.deptno
ORDER BY e.lname, e.fname, d.deptno
```

```
lname      fname      deptname   deptno
---------  ---------  ---------  ------
Brooke     Rachel     Marketing  4
Brown      Laura      IT         1
Lake       Leon       Research   3
River      Lilian     Research   3
Stream     Peter      Research   3
Taylor     Peter      NULL       NULL
Wood       Mike       IT         1
```

7.5 Join and Group By Together

1.

```
SELECT emp.lname, emp.empno,
   SUM(pe.hours_act) AS tot, COUNT(*) AS cnt   -- or COUNT(pe.projno)
FROM employee emp
   JOIN proj_emp pe
     ON emp.empno = pe.empno
GROUP BY emp.lname, emp.empno
ORDER BY emp.lname, emp.empno
```

```
LNAME      EMPNO        TOT        CNT
---------- -----  --------- ----------
Brooke     2245        1700          4
Lake       2345         100          1
Taylor     2884         500          2
Wood       2234         500          2
Brown      3546         400          1
Stream     2134         600          2
River      3547         300          1
```

2.

```
SELECT pr.projno, pr.project_name, SUM(pe.hours_act) AS hours_act,
       SUM(pe.hours_act) - SUM(pe.hours_est) AS diff
FROM project pr
  LEFT JOIN proj_emp pe
    ON pr.projno = pe.projno
WHERE priority IN (1, 3)
GROUP BY pr.projno, pr.project_name
```

```
PROJNO PROJECT_NAME   HOURS_ACT     DIFF
---- --------------- ---------- ----------
P3   WAREHOUSING            900        800
P2   BILLING                700        200
P5   CUSTOMERS             NULL       NULL
```

---

8 UNION

1.

```
SELECT city, empno AS id, 'EMPLOYEE' AS type
FROM employee
UNION ALL
SELECT location, projno, 'PROJECT'
FROM project
--ORDER BY city, id DESC
```

```
CITY            ID  TYPE
--------------- ---- --------
HELSINKI        2884 EMPLOYEE
HELSINKI        2245 EMPLOYEE
HELSINKI        P3   PROJECT
HELSINKI        P2   PROJECT
LONDON          2345 EMPLOYEE
LONDON          2234 EMPLOYEE
LONDON          2134 EMPLOYEE
LONDON          P4   PROJECT
LONDON          P1   PROJECT
SYDNEY          3547 EMPLOYEE
SYDNEY          3546 EMPLOYEE
SINGAPORE       P5   PROJECT
NULL            P6   PROJECT
```

9.1 Basic Subqueries

1.

```
SELECT project_name, priority
FROM project
WHERE priority =
(SELECT priority
 FROM project
 WHERE projno = 'P5')
```

```
PROJECT_NAME      PRIORITY
--------------- ----------
WAREHOUSING              3
CUSTOMERS                3
```

2.

```
SELECT lname, salary
FROM employee
WHERE salary =
(SELECT MAX(salary)
 FROM employee)
```

```
LNAME          SALARY
---------- ----------
Wood             3100
Brooke           3100
```

3.

```
SELECT project_name
FROM project
WHERE projno IN
(SELECT projno
 FROM proj_emp
 WHERE empno = '2134')

PROJECT_NAME
---------------
BOOKKEEPING
BILLING
```

4.

```
SELECT lname
FROM employee
WHERE empno IN
(SELECT empno
 FROM proj_emp
 WHERE projno IN
 (SELECT projno
  FROM project
  WHERE priority IN (1,3)))

LNAME
----------
Stream
Brooke
```

9.2 Correlated Subqueries – Connecting Subqueries to the Main Query

1.

```
SELECT deptno, lname, fname, start_date
FROM employee e1
WHERE start_date =
     (SELECT MIN(start_date)
      FROM employee e2
      WHERE e2.deptno = e1.deptno)
ORDER BY deptno

deptno     lname      fname      start_date
---------  ---------  ---------  -----------------------
1          Wood       Mike       2009-10-15 00:00:00.000
3          River      Lilian     2009-05-12 00:00:00.000
4          Brooke     Rachel     2014-09-24 00:00:00.000
```

EXISTS, p. 136

1.

```
SELECT project_name
FROM project
WHERE EXISTS
 (SELECT *
  FROM proj_emp
  WHERE proj_emp.projno = project.projno
  AND proj_emp.empno = '2134')

PROJECT_NAME
---------------
BOOKKEEPING
BILLING
```

2.

```
SELECT deptno, deptname
FROM department d
WHERE NOT EXISTS
  (SELECT *
   FROM employee e
   WHERE d.deptno = e.deptno)
ORDER BY deptno

deptno     deptname
---------- --------
2          Economy
```

10.2 “Does not Belong” Type Queries

1.

```
SELECT e.empno, e.lname, e.fname
FROM employee e
  LEFT JOIN  department d
        ON e.deptno = d.deptno
WHERE d.deptno IS NULL

empno      lname      fname
-------------    -------------     --------
2884       Taylor     Peter
```

11.3 Creating Tables, p. 176

1.

```
CREATE TABLE proj_hours
(projno CHAR (4) NOT NULL,
 project_name VARCHAR (15),       -- Oracle: VARCHAR2 (15)
 hours_sum INTEGER,
 PRIMARY KEY (projno))
```

2.

```
ALTER TABLE proj_hours
ADD priority SMALLINT
```

3.

SQL Server:

```
sp_rename 'proj_hours.hours_sum', 'hours_total'
```

Oracle, DB2, MySQL, PostgreSQL, Snowflake:

```
ALTER TABLE proj_hours RENAME COLUMN hours_sum TO hours_total
```

Hive:

```
ALTER TABLE proj_hours CHANGE hours_sum hours_total INTEGER
```

12.1 Adding Rows to a Table

1.

```
INSERT INTO project (projno, project_name, priority, location)
           VALUES   ('P7', 'APOLLO', NULL, 'NEW YORK')
```

or

```
INSERT INTO project (projno, project_name, location)
           VALUES   ('P7', 'APOLLO', 'NEW YORK')
```

2.

```
INSERT INTO proj_hours (projno, project_name, hours_total)
  SELECT p.projno, p.project_name, SUM(pe.hours_act)
  FROM project p LEFT JOIN proj_emp pe
    ON p.projno = pe.projno
  GROUP BY p.projno, p.project_name
```

12.2 Update

1.

```
UPDATE department
 SET deptname = 'IT Management'
 WHERE deptno = 1
```

2.

```
UPDATE project
SET priority = priority +1
WHERE location = 'HELSINKI'
```

12.3 Delete

1.

```
DELETE FROM proj_hours
WHERE projno = 'P2'
```

14.1 A View with one Table

1.

```
CREATE VIEW v_hels  AS
SELECT lname, fname, empno, city, start_date
FROM employee
WHERE city = 'HELSINKI'
```

2.

```
CREATE VIEW v_proj_hours (projno, project_name, acthours_tot)  AS
SELECT pr.projno, pr.project_name, SUM(pe.hours_act)
FROM project pr LEFT JOIN proj_emp pe
  ON pr.projno = pe.projno
GROUP BY pr.projno, pr.project_name
```

14.2 Complex Views

1.

```
CREATE VIEW v_proj_hours (projno, project_name, acthours_tot)  AS
SELECT p.projno, p.project_name, SUM(pe.hours_act)
FROM project p
   LEFT JOIN proj_emp pe
    ON p.projno = pe.projno
GROUP BY p.projno, p.project_name

Testing:
SELECT * FROM v_proj_hours
```

2.

```
CREATE VIEW v_projects AS
   SELECT p.project_name, p.location, e.lname, e.fname,
          e.salary, pe.hours_act
   FROM project p
     JOIN proj_emp pe
       ON p.projno = pe.projno
     JOIN employee e
       ON e.empno = pe.empno

SELECT project_name, lname, fname, hours_act
FROM v_projects
ORDER BY project_name, lname

SELECT lname, fname, project_name,  hours_act
FROM v_projects
ORDER BY lname, fname, project_name
```

18.1 NULL Handling, Conditional Reasoning and Converting

1.

```
SELECT project_name, COALESCE (priority, 0) AS priority,
               COALESCE (location,  'no location') AS location
FROM project
```

```
project_name          priority   location
------------------    ---------- ----------
BOOKKEEPING           2          LONDON
BILLING               2          HELSINKI
WAREHOUSING           4          HELSINKI
ACCOUNTING            2          LONDON
CUSTOMERS             3          SINGAPORE
STATISTICS            0          no location
```

CONDITIONAL REASONING WITH CASE

1.

```
SELECT project_name, priority,
CASE priority
           WHEN 1 then 'top'
           WHEN 2 then 'important'
           WHEN 3 then 'normal'
           ELSE 'don''t care'
END AS type
FROM project
ORDER BY project_name
```

```
project_name          priority   type
---------------       ---------- ---------
ACCOUNTING            2          important
BILLING               2          important
BOOKKEEPING           2          important
CUSTOMERS             3          normal
STATISTICS            NULL       don't care
WAREHOUSING           4          don't care
```

```
SELECT lname,
     CASE
         WHEN salary <= 2800 THEN salary
         ELSE 0
     END AS lowsal,
     CASE
         WHEN salary > 2800 THEN salary
         ELSE 0
     END AS highsal
FROM employee
ORDER BY lname
```

```
lname      lowsal     highsal
---------  ---------- --------
Brooke     0.00       3100.00
Brown      2650.00    0.00
Lake       2800.00    0.00
River      2800.00    0.00
Stream     2800.00    0.00
Taylor     0.00       2960.00
Wood       0.00       3100.00
```

CAST for Data Type Conversions, p 230

1.

```
SELECT lname, empno, start_date,
       CAST (empno AS INTEGER) + 1 AS empno_int,
              CAST (start_date AS CHAR(20)) AS start_date_char
FROM employee
ORDER BY lname
```

```
lname    empno start_date                empno_int  start_date_char
-------- ----- ------------------------  ---------  ------------------
Brooke   2245  2014-09-24 00:00:00.000   2246       Sep 24 2014 12:00AM
Brown    3546  2017-09-15 00:00:00.000   3547       Sep 15 2017 12:00AM
Lake     2345  2018-01-01 00:00:00.000   2346       Jan  1 2018 12:00AM
River    3547  2009-05-12 00:00:00.000   3548       May 12 2009 12:00AM
Stream   2134  2020-03-02 00:00:00.000   2135       Mar  2 2020 12:00AM
Taylor   2884  2009-05-12 00:00:00.000   2885       May 12 2009 12:00AM
Wood     2234  2009-10-15 00:00:00.000   2235       Oct 15 2009 12:00AM
```

You may see the date in a different format in your product.

19.1 A Subquery as a Table, Derived Table

1.

```
SELECT e.lname, e.fname, e.city, e.salary, subq.avgsalary
 FROM employee e
   JOIN
     (SELECT city, AVG(salary) avgsalary
      FROM employee
       GROUP BY city) subq
    ON e.city = subq.city
WHERE e.salary <= subq.avgsalary
```

```
lname      fname      city       salary     avgsalary
--------   ---------  ---------  ---------  -----------
Taylor     Peter      HELSINKI   2960.00    3030.000000
Lake       Leon       LONDON     2800.00    2900.000000
Stream     Peter      LONDON     2800.00    2900.000000
Brown      Laura      SYDNEY     2650.00    2725.000000
```

19.3 Outer Join – Some Deeper Analysis

1.

```
SELECT  e.empno, e.lname, e.fname, e.salary, d.deptno, d.deptname
FROM  employee e
LEFT JOIN
     department d
ON d.deptno = e.deptno
AND (d.deptname = 'Research' OR d.deptname = 'Marketing')
ORDER BY e.lname , e.fname
```

empno	lname	fname	salary	deptno	deptname
2245	Brooke	Rachel	3100.00	4	Marketing
3546	Brown	Laura	2650.00	NULL	NULL
2345	Lake	Leon	2800.00	3	Research
3547	River	Lilian	2800.00	3	Research
2134	Stream	Peter	2800.00	3	Research
2884	Taylor	Peter	2960.00	NULL	NULL
2234	Wood	Mike	3100.00	NULL	NULL

OR

```
SELECT  e.empno, e.lname, e.fname, e.salary, q.deptno, q.deptname
FROM  employee e
LEFT JOIN
   (SELECT deptno, deptname
   FROM department d
   WHERE d.deptname = 'Research' OR d.deptname = 'Marketing') q
ON q.deptno = e.deptno
ORDER BY e.lname , e.fname
```

In both queries a good alternative for the deptname condition would be:

```
d.deptname = IN ('Research', 'Marketing')
```

19.9 "Most of" Queries

1.

```
SELECT city, COUNT(*) AS cnt
FROM employee
GROUP BY city
HAVING COUNT(*) >= ALL
       (SELECT COUNT(*)
        FROM employee
        GROUP BY city)
```

city	cnt
LONDON	3

```
Or:

WITH city_cnt
AS
  (SELECT city, COUNT(*) AS cnt
   FROM employee
   GROUP BY city)
SELECT city, cnt
FROM city_cnt
WHERE cnt =
      (SELECT MAX(cnt)
       FROM city_cnt)
```

20.5 Fetch Top n

```
1.

SQL Server:

SELECT TOP 3 lname, fname, tax_rate
FROM employee
ORDER BY tax_rate DESC

lname   fname      tax_rate
------  ---------  --------
River   Lilian     37.0
Wood    Mike       33.0
Brooke  Rachel     31.0

DB2, Snowflake, PostgreSQL, Oracle:

SELECT lname, fname, tax_rate
FROM employee
ORDER BY tax_rate DESC
FETCH FIRST 3 ROWS ONLY


Snowflake, MySQL, PostgreSQL and Hive:

SELECT lname, fname,  tax_rate
FROM employee
ORDER BY tax_rate DESC
LIMIT 3
```

2.

```
SELECT *
FROM
   (SELECT lname, fname, tax_rate,
    DENSE_RANK() OVER (ORDER BY tax_rate DESC ) AS ord
    FROM employee) s
WHERE s.ord <= 3
```

```
lname   fname      tax_rate   ord
------  ---------  ---------  ---
River   Lilian     37.0       1
Wood    Mike       33.0       2
Brooke  Rachel     31.0       3
Taylor  Peter      31.0       3
```

20.9 Detail and Sum on the Same Row

1.

SQL Server:

```
SELECT emp.lname, emp.fname, emp.city, emp.salary,
        CAST(salary*100/csum.sum_salary AS  DECIMAL(7,2)) AS share,
        csum.sum_salary
FROM employee emp
JOIN     (SELECT city, SUM(salary) AS sum_salary
          FROM employee
          GROUP BY city) csum
ON  emp.city = csum.city
ORDER BY emp.lname
```

```
lname   fname      city       salary     share      sum_salary
------- ---------  ---------  ---------  ---------- -----------
Brooke  Rachel     HELSINKI   3100.00    51.16      6060.00
Brown   Laura      SYDNEY     2650.00    48.62      5450.00
Lake    Leon       LONDON     2800.00    32.18      8700.00
River   Lilian     SYDNEY     2800.00    51.38      5450.00
Stream  Peter      LONDON     2800.00    32.18      8700.00
Taylor  Peter      HELSINKI   2960.00    48.84      6060.00
Wood    Mike       LONDON     3100.00    35.63      8700.00
```

Oracle, PostgreSQL, Snowflake, Hive, MySQL:

```
SELECT emp.lname, emp.fname, emp.city, emp.salary,
       ROUND(emp.salary * 100 / csum.sum_salary, 2) AS sal_share,
       csum.sum_salary
FROM employee emp
JOIN (SELECT city, SUM(salary) AS sum_salary
      FROM employee
      GROUP BY city) csum
ON emp.city = csum.city
```

Pleas note that in Oracle “share” is a reserved word and cannot be used as an alias, so we changed it ot “sal_share”.

DB2:

```
SELECT emp.lname, emp.fname, emp.city, emp.salary,
        DECIMAL(salary*100/ csum.sum_salary, 7,2) AS share,
        csum.sum_salary
FROM employee emp
JOIN    (SELECT city, SUM(salary) AS sum_salary
          FROM employee
          GROUP BY city) csum
ON  emp.city = csum.city
```