Self-test SQL Workshop

Document: e0087test.fm

05/01/2016

ABIS Training & Consulting
P.O. Box 220
B–3000 Leuven
Belgium
INTRODUCTION TO THE SELF-TEST SQL WORKSHOP

Instructions

The aim of this test is to check if you have a thorough knowledge of SQL. After the test you will know whether the 2-day SQL workshop is still worthwhile for you to follow, or whether you can immediately enrol for the 1-day SQL for advanced users.

This test consists of 15 (mainly multiple choice). Sometimes multiple answers must be given in which case this will clearly be indicated. Write down your responses and compare them with the solutions given on the last page. This test will take about half an hour.

Remarks:

- The SQL syntax used in the questions is ANSI/ISO SQL:2003, currently supported by all platforms (DB2, Oracle, MySQL, SQL Server, PostgreSQL, ...)
- This is an advanced SQL test. If you are unsure about your basic SQL skills, you should better start with the self-test for “SQL and RDBMS: Fundamentals course”.

Table and column information

The questions are based on the following tables:

- COURSES: describes all the courses that can be organised.
- SESSIONS: describes courses organised at a certain moment.
- PERSONS: describes all persons (instructors, enrollees, other persons).
- ENROLMENTS: contains all information on enrollees and their sessions.

The following relations exist between the tables:

- COURSES - SESSIONS
  - obtain course information for a certain session
  - COURSES.CID = SESSIONS.S_CID
- SESSIONS - ENROLMENTS
  - obtain session information for a certain enrolment
  - SESSIONS.SNO = ENROLMENTS.E_SNO
- PERSONS - SESSIONS
  - obtain person information for the instructor of a certain session
  - PERSONS.PNO = SESSIONS.SINS_PNO
- PERSONS - ENROLMENTS
  - obtain person information for a certain enrollee
  - PERSONS.PNO = ENROLMENTS.E_PNO
Table content and column descriptions

- **COURSES** table
  
<table>
<thead>
<tr>
<th>CID</th>
<th>CTITLE</th>
<th>CDUR</th>
</tr>
</thead>
<tbody>
<tr>
<td>7890</td>
<td>DB2</td>
<td>5</td>
</tr>
<tr>
<td>7910</td>
<td>Unix</td>
<td>4</td>
</tr>
<tr>
<td>8500</td>
<td>Oracle</td>
<td>5</td>
</tr>
<tr>
<td>8000</td>
<td>SQLServer</td>
<td>5</td>
</tr>
<tr>
<td>9000</td>
<td>SQL workshop</td>
<td>3</td>
</tr>
</tbody>
</table>

  - **CID**: required, alphanumeric: course number (primary key)
  - **CTITLE**: required, alphanumeric: course title
  - **CDUR**: required, numeric: course duration (in days).

- **SESSIONS** table (8 rows)
  
<table>
<thead>
<tr>
<th>SNO</th>
<th>S_CID</th>
<th>SDATE</th>
<th>SINS_PNO</th>
<th>SCANCEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>7890</td>
<td>2015-12-02</td>
<td>3</td>
<td>(NULL)</td>
</tr>
<tr>
<td>11</td>
<td>7910</td>
<td>2015-11-04</td>
<td>1</td>
<td>(NULL)</td>
</tr>
<tr>
<td>12</td>
<td>7890</td>
<td>2016-01-08</td>
<td>3</td>
<td>C</td>
</tr>
<tr>
<td>13</td>
<td>7890</td>
<td>2016-02-02</td>
<td>3</td>
<td>(NULL)</td>
</tr>
<tr>
<td>14</td>
<td>8000</td>
<td>2016-04-05</td>
<td>2</td>
<td>C</td>
</tr>
<tr>
<td>15</td>
<td>7910</td>
<td>2016-01-08</td>
<td>36</td>
<td>C</td>
</tr>
<tr>
<td>16</td>
<td>8500</td>
<td>2016-04-05</td>
<td>36</td>
<td>(NULL)</td>
</tr>
<tr>
<td>17</td>
<td>9000</td>
<td>2016-06-07</td>
<td>36</td>
<td>(NULL)</td>
</tr>
</tbody>
</table>

  - **SNO**: required, numeric: session number (primary key).
  - **S_CID**: optional, alphanumeric: course number (foreign key to COURSES).
  - **SDATE**: optional: start date of the session.
  - **SINS_PNO**: required, numeric: session instructor (foreign key to PERSONS).
  - **SCANCEL**: optional: indicates if the session is cancelled ("C" means cancelled, empty (NULL) means not cancelled).
### PERSONS table (19 rows)

<table>
<thead>
<tr>
<th>PNO</th>
<th>PNAME</th>
<th>P_CONO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SMITHE</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>TAUERNIER</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>DE KEYSER</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>HEBBELYNCK</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>VAN DE BROECK</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>VAN HEIJKOOP</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>DE WINDT</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>SPENGER</td>
<td>10</td>
</tr>
<tr>
<td>9</td>
<td>BENOIT</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>BENOIT</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>LOOSE</td>
<td>(NULL)</td>
</tr>
<tr>
<td>13</td>
<td>PARKER</td>
<td>6</td>
</tr>
<tr>
<td>15</td>
<td>DEHEM</td>
<td>7</td>
</tr>
<tr>
<td>17</td>
<td>PIELAGE</td>
<td>4</td>
</tr>
<tr>
<td>18</td>
<td>GELADE</td>
<td>2</td>
</tr>
<tr>
<td>33</td>
<td>BUENK</td>
<td>9</td>
</tr>
<tr>
<td>36</td>
<td>ADAMSON</td>
<td>8</td>
</tr>
<tr>
<td>45</td>
<td>MOORS</td>
<td>4</td>
</tr>
<tr>
<td>50</td>
<td>MAK</td>
<td>(NULL)</td>
</tr>
</tbody>
</table>

- **PNO**: required, numeric: person number (primary key).
- **PNAME**: optional, alphanumeric: name.
- **P_CONO**: optional, numeric: number of the company the person works for.

### ENROLMENTS table (14 rows, 9 different enrollees)

<table>
<thead>
<tr>
<th>E_SNO</th>
<th>E_PNO</th>
<th>ECANCEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>4</td>
<td>(NULL)</td>
</tr>
<tr>
<td>10</td>
<td>7</td>
<td>C</td>
</tr>
<tr>
<td>11</td>
<td>45</td>
<td>(NULL)</td>
</tr>
<tr>
<td>11</td>
<td>13</td>
<td>(NULL)</td>
</tr>
<tr>
<td>12</td>
<td>4</td>
<td>(NULL)</td>
</tr>
<tr>
<td>13</td>
<td>15</td>
<td>C</td>
</tr>
<tr>
<td>13</td>
<td>36</td>
<td>(NULL)</td>
</tr>
<tr>
<td>14</td>
<td>3</td>
<td>(NULL)</td>
</tr>
<tr>
<td>14</td>
<td>18</td>
<td>C</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>(NULL)</td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>(NULL)</td>
</tr>
<tr>
<td>15</td>
<td>7</td>
<td>(NULL)</td>
</tr>
<tr>
<td>16</td>
<td>3</td>
<td>(NULL)</td>
</tr>
<tr>
<td>16</td>
<td>18</td>
<td>(NULL)</td>
</tr>
</tbody>
</table>

- **E_SNO**: required, numeric: session number for the enrolment (foreign key to SESSIONS) (primary key together with E_PNO)
- **E_PNO**: required, numeric: the enrollee (foreign key to PERSONS) (primary key together with E_SNO)
- **ECANCEL**: optional: “C” when enrolment was cancelled, NULL if not cancelled.
QUESTIONS  SELF-TEST SQL WORKSHOP

1. Which ones of the following queries produce exactly 1 result row? [2 correct answers.]

   [ ] [a]
   
   ```sql
   SELECT COUNT(*)
   FROM   PERSONS
   WHERE  PNO > 100
   ```

   [ ] [b]
   
   ```sql
   SELECT PNO, COUNT(*)
   FROM   PERSONS
   WHERE  PNO = 2
   ```

   [ ] [c]
   
   ```sql
   SELECT COUNT(*)
   FROM   PERSONS
   GROUP BY PNO
   ```

   [ ] [d]
   
   ```sql
   SELECT PNAME
   FROM   PERSONS INNER JOIN SESSIONS ON PNO = SINS_PNO
   WHERE  PNO = 36
   ```

   [ ] [e]
   
   ```sql
   SELECT PNAME
   FROM   PERSONS LEFT OUTER JOIN ENROLMENTS ON PNO = E_PNO
   WHERE  PNO = 2
   GROUP BY PNAME
   ```

   [ ] [f]
   
   ```sql
   SELECT SUM(CDUR)
   FROM   COURSES, SESSIONS, ENROLMENTS
   WHERE  CID = S_CID AND SNO = E_SNO
   GROUP BY CID
   ```

2. How many result rows are produced by this query?

   ```sql
   SELECT E_SNO
   FROM   ENROLMENTS
   UNION
   SELECT SNO
   FROM   SESSIONS
   WHERE  SNO BETWEEN 15 AND 17
   ```

   Answer: ..........

   8 unique values from the first query, and additionally the value 17 from the second query. "UNION" will remove duplicates, even when the all come from the first query.
3. Which queries produce the following table as the result? [3 correct answers.]

<table>
<thead>
<tr>
<th>PNO</th>
<th>PNAME</th>
<th>ENROLLEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SMITHS</td>
<td>ENROLLEE</td>
</tr>
<tr>
<td>3</td>
<td>DE KEYSER</td>
<td>ENROLLEE</td>
</tr>
<tr>
<td>4</td>
<td>HEBBEYNCK</td>
<td>ENROLLEE</td>
</tr>
<tr>
<td>7</td>
<td>DE WINDT</td>
<td>ENROLLEE</td>
</tr>
<tr>
<td>13</td>
<td>PARKER</td>
<td>ENROLLEE</td>
</tr>
<tr>
<td>15</td>
<td>DEHEM</td>
<td>ENROLLEE</td>
</tr>
<tr>
<td>18</td>
<td>GELADE</td>
<td>ENROLLEE</td>
</tr>
<tr>
<td>36</td>
<td>ADAMSON</td>
<td>ENROLLEE</td>
</tr>
<tr>
<td>45</td>
<td>MOORS</td>
<td>ENROLLEE</td>
</tr>
<tr>
<td>1</td>
<td>SMITHS</td>
<td>INSTRUCTOR</td>
</tr>
<tr>
<td>2</td>
<td>TAVERNIER</td>
<td>INSTRUCTOR</td>
</tr>
<tr>
<td>3</td>
<td>DE KEYSER</td>
<td>INSTRUCTOR</td>
</tr>
<tr>
<td>36</td>
<td>ADAMSON</td>
<td>INSTRUCTOR</td>
</tr>
</tbody>
</table>

[] [a]

```sql
SELECT PNO, PNAME, 'ENROLLEE OR INSTRUCTOR'
FROM PERSONS INNER JOIN SESSIONS ON PNO = SINS_PNO
    INNER JOIN ENROLMENTS ON PNO = E_PNO
ORDER BY 3, 1
```

[] [b]

```sql
SELECT PNO, PNAME, CASE PNO WHEN E_PNO THEN 'ENROLLEE' ELSE 'INSTRUCTOR' END
FROM PERSONS INNER JOIN SESSIONS ON PNO = SINS_PNO
    INNER JOIN ENROLMENTS ON PNO = E_PNO
ORDER BY 3, 1
```

[] [c]

```sql
SELECT PNO, PNAME, 'INSTRUCTOR'
FROM PERSONS
WHERE PNO IN (SELECT SINS_PNO
                   FROM SESSIONS)
UNION ALL
SELECT PNO, PNAME, 'ENROLLEE'
FROM PERSONS INNER JOIN ENROLMENTS ON PNO = E_PNO
ORDER BY 3, 1
```

[] [d]

```sql
SELECT DISTINCT PNO, PNAME, 'INSTRUCTOR'
FROM PERSONS INNER JOIN SESSIONS ON PNO = SINS_PNO
UNION ALL
SELECT PNO, PNAME, 'ENROLLEE'
FROM PERSONS
WHERE PNO IN (SELECT E_PNO
                   FROM ENROLMENTS)
ORDER BY 3, 1
```

[] [e]

```sql
SELECT PNO, PNAME, 'INSTRUCTOR'
```
FROM PERSONS INNER JOIN SESSIONS ON PNO = SINS_PNO
UNION
SELECT PNO, PNAME, 'ENROLLEE'
FROM PERSONS
WHERE PNO IN (SELECT E_PNO
               FROM ENROLMENTS)
ORDER BY 3, 1

[] [f]

SELECT DISTINCT PNO, PNAME, 'INSTRUCTOR'
FROM PERSONS INNER JOIN SESSIONS ON PNO = SINS_PNO
UNION
SELECT PNO, PNAME, 'ENROLLEE'
FROM PERSONS P
WHERE EXISTS (SELECT E_PNO
               FROM ENROLMENTS
               WHERE E_PNO = P.PNO)
ORDER BY 3, 1

4. **How many result rows are produced by the following query?**

   SELECT DISTINCT PNO
   FROM PERSONS LEFT OUTER JOIN ENROLMENTS ON PNO = E_PNO

   **Answer:** ..........

   The number of rows of PERSONS (because "left outer join")
5. Which queries give an answer to the following question? [2 correct answers.]

   *Give the number of all sessions for which none of the enrolments have been cancelled.*

   [a] 
   ```sql
   SELECT DISTINCT SNO
   FROM SESSIONS, ENROLMENTS
   WHERE SNO = E_SNO AND ECANCEL IS NULL
   ```
   
   [b] 
   ```sql
   SELECT DISTINCT SNO
   FROM SESSIONS, ENROLMENTS
   WHERE SNO = E_SNO AND ECANCEL IS NOT NULL
   ```
   
   [c] 
   ```sql
   WITH E AS (SELECT E_SNO
               FROM ENROLMENTS
               WHERE ECANCEL IS NOT NULL)
   SELECT SNO
   FROM SESSIONS LEFT OUTER JOIN E ON SNO = E_SNO
   WHERE E_SNO IS NULL
   ```
   
   [d] 
   ```sql
   SELECT SNO
   FROM SESSIONS
   WHERE SNO IN (SELECT E_SNO
                  FROM ENROLMENTS
                  WHERE ECANCEL IS NULL)
   ```
   
   [e] 
   ```sql
   SELECT SNO FROM SESSIONS
   EXCEPT -- or MINUS when using Oracle
   SELECT E_SNO FROM ENROLMENTS WHERE ECANCEL IS NOT NULL
   ```
   
   [f] 
   ```sql
   SELECT SNO
   FROM SESSIONS S
   WHERE NOT EXISTS (SELECT 1
                      FROM ENROLMENTS
                      WHERE E_SNO = S.SNO AND ECANCEL IS NOT NULL)
   ```
   
   [g] 
   ```sql
   SELECT SNO
   FROM SESSIONS INNER JOIN ENROLMENTS ON SNO = E_SNO
   WHERE ECANCEL IS NULL
   ```
   
   [h] 
   ```sql
   SELECT SNO
   FROM SESSIONS INNER JOIN ENROLMENTS ON SNO = E_SNO
   WHERE ECANCEL IS NOT NULL
   ```
6. Which queries produce the following table “all enrollees”? [3 correct answers.]

<table>
<thead>
<tr>
<th>PNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMITHS</td>
</tr>
<tr>
<td>DE KEYSER</td>
</tr>
<tr>
<td>HEBBELYNCK</td>
</tr>
<tr>
<td>VAN HEIJKOOP</td>
</tr>
<tr>
<td>DE WINDT</td>
</tr>
<tr>
<td>PARKER</td>
</tr>
<tr>
<td>DEHEM</td>
</tr>
<tr>
<td>GELADE</td>
</tr>
<tr>
<td>MOORS</td>
</tr>
</tbody>
</table>

- [ ] [a]  
  ```sql
  SELECT PNAME, FROM PERSONS  
  WHERE PNO IN (SELECT E_PNO AS PNO FROM ENROLMENTS)  
  ```

- [ ] [b]  
  ```sql
  SELECT PNAME  
  FROM PERSONS INNER JOIN ENROLMENTS ON PNO = E_PNO  
  ```

- [ ] [c]  
  ```sql
  SELECT PNAME, FROM PERSONS  
  WHERE PNO = ANY (SELECT E_PNO FROM ENROLMENTS)  
  ```

- [ ] [d]  
  ```sql
  SELECT PNAME, FROM PERSONS  
  WHERE EXISTS (SELECT E_PNO FROM ENROLMENTS)  
  ```

- [ ] [e]  
  ```sql
  SELECT PNAME  
  FROM (SELECT E_PNO FROM ENROLMENTS WHERE E_PNO IS NOT NULL) E  
  INNER JOIN PERSONS ON PNO = E.E_PNO  
  ```

- [ ] [f]  
  ```sql
  SELECT PNAME  
  FROM PERSONS LEFT OUTER JOIN ENROLMENTS ON PNO = E_PNO  
  GROUP BY PNAME  
  ```

- [ ] [g]  
  ```sql
  SELECT PNAME  
  FROM PERSONS RIGHT OUTER JOIN ENROLMENTS ON PNO = E_PNO  
  GROUP BY PNAME  
  ```
7. Which question corresponds best to the following query?

```
SELECT P_CONO, COUNT(*)
FROM PERSONS P
WHERE EXISTS (SELECT SNO
              FROM SESSIONS
              WHERE SINS_PNO = P.PNO)
GROUP BY P_CONO
```

- (a) Give per instructor the number of sessions he teaches. Give also the company where he is employed.
- (b) Give per company the number of employees who followed a course.
- (c) Give the number of sessions per course, and also the company where the instructor is employed.
- (d) Give the number of instructors per company.
8. Which ones of the following queries are equivalent to this query? [2 correct answers.]

```
SELECT PNAME
FROM PERSONS
WHERE PNO = ( SELECT MAX(PNO) FROM PERSONS )
```

[ ] [a]

```
SELECT PNAME
FROM PERSONS
WHERE PNO >= ANY (SELECT PNO FROM PERSONS)
```

[ ] [b]

```
SELECT PNAME
FROM PERSONS
WHERE PNO >= ALL (SELECT PNO FROM PERSONS)
```

[ ] [c]

```
SELECT PNAME
FROM PERSONS P1
WHERE EXISTS (SELECT MAX(PNO)
              FROM PERSONS P2
              WHERE P1.PNO = P2.PNO)
```

[ ] [d]

```
SELECT PNAME, MAX(PNO)
FROM PERSONS
GROUP BY PNAME
```

[ ] [e]

```
SELECT P1.PNAME
FROM PERSONS P1
LEFT OUTER JOIN
    PERSONS P2      ON P1.PNO < P2.PNO
GROUP BY P1.PNO, P1.PNAME
HAVING COUNT(P2.PNO) = 0
```

9. Which queries give an answer to the following question? [3 correct answers.]

*Give the list of all courses, also those for which no session has been planned yet. Give also all corresponding session numbers and the date on which each session starts.*

[ ] [a]

```
SELECT CID, SNO, SDATE
FROM COURSES INNER JOIN SESSIONS ON CID = S_CID
```

[ ] [b]

```
SELECT S_CID, SNO, SDATE
FROM SESSIONS
```
[c]  
```
SELECT CID, SNO, SDATE  
FROM COURSES LEFT OUTER JOIN SESSIONS ON CID = S_CID
```

[d]  
```
SELECT CID, SNO, SDATE  
FROM COURSES RIGHT OUTER JOIN SESSIONS ON CID = S_CID
```

[e]  
```
WITH S AS (SELECT SNO, S_CID, SDATE  
            FROM SESSIONS  
            WHERE S_CID IS NOT NULL)  
SELECT CID, SNO, SDATE  
FROM COURSES INNER JOIN S ON CID = S_CID  
UNION ALL  
SELECT CID, 0, CAST(NULL AS DATE)  
FROM COURSES  
WHERE CID NOT IN (SELECT S_CID FROM S)
```

[f]  
```
SELECT C.CID, S.SNO, S.SDATE  
FROM (SELECT CID FROM COURSES) C  
LEFT OUTER JOIN  
   (SELECT SNO, S_CID, SDATE FROM SESSIONS) S  
on S_CID = CID
```

[g]  
```
SELECT CID, SNO, SDATE  
FROM COURSES INNER JOIN SESSIONS ON CID = S_CID  
UNION ALL  
SELECT S_CID, SNO, SDATE  
FROM SESSIONS  
WHERE S_CID IS NULL
10. Which query implements the following question?

_Give the names of all instructors who have in addition also followed more than 1 course._

(a) 

```
SELECT PNAME  FROM  PERSONS
WHERE  PNO IN (SELECT E_PNO
FROM   ENROLMENTS INNER JOIN SESSIONS ON E_SNO = SNO
WHERE  E_PNO = SINS_PNO
    AND ECANCEL IS NULL
    AND SCANCEL IS NULL
GROUP BY E_PNO  HAVING COUNT(*) > 1)
```

The subquery lists all instructors who were enrolled for their own sessions.

(b) 

```
SELECT PNAME  FROM PERSONS
WHERE  PNO IN (SELECT SINS_PNO
FROM   SESSIONS
WHERE  SCANCEL IS NULL
    AND SNO IN (SELECT E_SNO
    FROM   ENROLMENTS
    WHERE  ECANCEL IS NULL
    GROUP BY E_PNO  HAVING COUNT(*) > 1))
```

Syntax error in the innermost subquery: E_SNO not guaranteed constant in a group.

GROUP BY E_SNO instead of E_PNO would list all instructors of sessions having more than one not cancelled enrolment.

(c) 

```
SELECT PNAME
FROM   PERSONS INNER JOIN
(SELECT E_PNO FROM ENROLMENTS
WHERE  ECANCEL IS NULL
    AND E_SNO IN (SELECT SNO FROM SESSIONS  WHERE SCANCEL IS NULL)
GROUP BY E_PNO  HAVING COUNT(*) > 1)   E
ON E_PNO = PNO
WHERE  PNO IN (SELECT SINS_PNO FROM SESSIONS)
```

GROUP BY PNAME would be less accurate, in case the of two instructors with the same name.

(d) 

```
SELECT PNAME
FROM   PERSONS INNER JOIN ENROLMENTS  ON PNO = E_PNO
    INNER JOIN SESSIONS S1 ON E_SNO = S1.SNO
    INNER JOIN SESSIONS S2 ON PNO = S1.SINS_PNO
WHERE  S1.SCANCEL IS NULL  AND  ECANCEL IS NULL
GROUP BY E_PNO, PNAME  HAVING COUNT(*) > 1
```

Instructors who gave more than one session, but only followed a single session themselves, are nonetheless listed.

(e) 

```
WITH P AS (SELECT PNO, PNAME FROM PERSONS),
    E AS (SELECT E_PNO, E_SNO FROM ENROLMENTS WHERE ECANCEL IS NULL),
    S AS (SELECT SNO, SINS_PNO FROM SESSIONS  WHERE SCANCEL IS NULL)
SELECT PNAME
FROM   P INNER JOIN E    ON PNO = E_PNO
    INNER JOIN S S1 ON E_SNO = S1.SNO
    INNER JOIN S S2 ON PNO = S2.SINS_PNO
GROUP BY E_PNO, PNAME  HAVING COUNT(*) > 1
```

Instructors who gave more than one session, but only followed a single session themselves, are nonetheless listed.
11. What can be said about this query?

```
SELECT SNO, PNAME, SDATE
FROM   SESSIONS, PERSONS
WHERE  SINS_PNO = PNO
UNION
SELECT E_PNO, PNAME
FROM   PERSONS, ENROLMENTS
WHERE  PNO = E_PNO
ORDER BY 1
```

O (a) Query cannot be executed (gives a syntax error).
O (b) Query can be executed and makes sense (according to the table and column definitions).
O (c) Query can be executed but returns nonsense.

12. What can be said about this query?

```
SELECT SNO, SDATE, PNAME
FROM   SESSIONS INNER JOIN ENROLMENTS ON SNO = E_SNO
       INNER JOIN PERSONS ON P_CONO = E_PNO
WHERE  ECANCEL IS NULL
   AND  SCANCEL IS NULL
```

O (a) Query cannot be executed (gives a syntax error).
O (b) Query can be executed and makes sense (according to the table and column definitions).
O (c) Query can be executed but returns nonsense.

P_CONO & E_PNO don't have the same meaning, even though syntactically there's nothing wrong.

13. What can be said about this query?

```
SELECT SNO, SDATE, S_CID
FROM   SESSIONS S
WHERE  SCANCEL IS NOT NULL
   AND  SDATE = (SELECT MAX(SDATE)
                 FROM   SESSIONS
                 WHERE  S_CID = S.S_CID)
```

O (a) Query cannot be executed (gives a syntax error).
O (b) Query can be executed and makes sense (according to the table and column definitions).

This query lists the last session of each course, including date and course number.
O (c) Query can be executed but returns nonsense.

14. What can be said about this query?

```
SELECT PNAME, COUNT(*)
FROM   PERSONS INNER JOIN SESSIONS ON SINS_PNO = PNO
```

O (a) Query cannot be executed (gives a syntax error).
O (b) Query can be executed and makes sense (according to the table and column definitions).
O (c) Query can be executed but returns nonsense.
15. What can be said about this query?

```
SELECT (SELECT COUNT(*) AS nr_8000
        FROM   SESSIONS
        WHERE  S_CID = '8000')
   * 100.0 / COUNT(S_CID) AS percent_8000
FROM   SESSIONS
```

O (a) Query cannot be executed (gives a syntax error).

O (b) Query can be executed and makes sense (according to the table and column definitions).

O (c) Query can be executed but returns nonsense.
EVALUATION.

Here are the correct answers to all questions:

1. a e
2. 8
3. d e f
4. 19
5. c e f
6. a c g
7. d
8. b e
9. c e f
10. c
11. a
12. c
13. b
14. a
15. b

Give yourself 1 point for each correctly answered question; for multiple answer questions, all answers should be correct.

When your score is 80% or above, you are ready for our Advanced SQL course.

When your score is between 50% and 80%, following the course SQL workshop will allow you to refine your SQL knowledge.

When your score is less than 50%, following the course SQL workshop is advisable. You will get a high return from this course. Be sure your basic SQL knowledge is sufficient: fill out the corresponding self-test (see PDF file) to verify this.