Course report 2026

Higher Computing Science, SCQF level 6

We've written this report for teachers and lecturers. We recommend you read this with the published course assessment documents and marking instructions.

The report:

- provides a summary of how each course component performed
- highlights areas of the question paper and coursework where candidates performed well and areas where they did not perform well
- has advice you might find useful when preparing candidates for assessment

We collect and prepare the statistics in course reports before we complete the appeals process.

Copyright © Qualifications Scotland 2026

Statistical and grade boundary information

Course entries

Number of resulted entries in 2025: 3,962

Number of resulted entries in 2026: 3,989

Distribution of course awards (including minimum mark to achieve each grade)

Course award	Number of candidates	Percentage	Cumulative percentage	Minimum mark required
A	1,805	45.2	45.2	84
B	708	17.7	63.0	72
C	608	15.2	78.2	60
D	447	11.2	89.4	48
No award	421	10.6	100	Not applicable

We've not applied rounding to these statistics.

In this report:

- 'Most' means greater than or equal to 70%.
- 'Many' means 50% to 69%.
- 'Some' means 25% to 49%.
- 'Few' or 'a few' means less than 25%.

You can find statistical reports on our website.

General commentary on grade boundaries

Our main aim when setting grade boundaries is to be fair to candidates across all subjects and levels, and to maintain comparable standards across the years, even as arrangements evolve and change.

For most National Courses, we aim to set examinations and other external assessments and create marking instructions that allow:

- a competent candidate to score a minimum of 50% of the available marks (the notional grade C boundary)
- a well-prepared, very competent candidate to score at least 70% of the available marks (the notional grade A boundary)

It's very challenging to get the standard on target every year, in every subject, at every level. Therefore, we hold a grade boundary meeting for each course to bring together all the information available (statistical and qualitative) and make final decisions on grade boundaries based on this information. Senior Qualifications Scotland staff normally chair these meetings.

Principal assessors use their subject expertise to evaluate the performance of the assessment and propose suitable grade boundaries based on the full range of evidence. We can adjust the grade boundaries as a result of the discussion at these meetings. This allows the pass rate to be unaffected in circumstances where there is evidence that the question paper or other assessment has been more, or less, difficult than usual.

- We can adjust the grade boundaries downwards if there's evidence that the question paper or other assessment has been more difficult than usual.
- We can adjust the grade boundaries upwards if there's evidence that the question paper or other assessment has been less difficult than usual.
- If levels of difficulty are comparable to previous years, we maintain similar grade boundaries.

Every year, we evaluate the performance of our assessments in a fair way, while ensuring standards are maintained so that our qualifications remain credible. To do

this, we measure evidence of candidates' knowledge and skills against the national standard.

For full details of the approach, please refer to the Awarding and Grading for National Courses Policy.

How the course assessment performed

Question paper and assignment

Marker feedback suggested that the question paper and assignment were fair and accessible for candidates. This was supported by statistical data which demonstrated that, overall, both assessments performed as intended. As a result, we did not make any adjustments to grade boundaries.

In both the question paper and the assignment, 66% candidates completed the ‘Database design and development’ section, and 34% of candidates completed the ‘Web design and development’ section.

How candidates performed

Question paper

Section 1: Software design and development, and computer systems

Data representation

Candidates continued to perform strongly in data representation and demonstrated a good understanding of number representation and conversion. Many candidates accurately converted an 8-bit two's complement binary number into denary and determined the number of bits required to represent a value using two's complement notation (questions 1(a) and (b)). Candidates also showed a good understanding of floating-point representation, with most converting a binary number to floating-point form (question 2(a)). While many candidates showed an awareness of mantissa and exponent when converting, few described how to reallocate the number of bits allocated to both to increase the precision of the floating-point numbers. Many candidates instead gave generic answers (question 2(b)).

Another area of strength was character representation, with many candidates recognising that an advantage of Unicode over ASCII is its ability to represent a larger number of characters or languages (question 9(c)).

Security risks, precautions and environmental impact

Candidates answered questions requiring an understanding of security risks and precautions, particularly those related to privacy, less confidently. Candidates did not clearly state the name of the domain or suggest that the name of the owner was in the digital certificate (question 6(b)). They also often confused the role of a digital certificate and the role of a digital signature (question 6(c)). When discussing tracking cookies, most candidates incorrectly stated the security risk of cookies (question 9(d)).

Most candidates, when describing the environmental benefits of traffic control systems, did not refer to any technological aspect of the intelligent system (question 6(d)).

Analysis and design

Many candidates demonstrated a good understanding of the analysis stage of the software development process and identified an appropriate boundary for a program from a given scenario (question 3). Candidates also demonstrated effective problem-solving skills and most candidates effectively designed a suitable algorithm that included file access. However, in the algorithm design question, candidates frequently omitted the television show criteria (question 9(b)).

Data flow continues to be an area that candidates find demanding. Many candidates did not gain full marks because they either answered using parallel arrays instead of an array of records, or passed unnecessary additional parameters (question 10(b)).

Implementation and evaluation

Candidate performance on questions assessing aptitude on programming constructs and program implementation was encouraging. Many candidates demonstrated that they had a good grasp of programming constructs and could accurately identify a formal parameter and its corresponding actual parameter (question 3). Most candidates showed a good understanding of variables and their scope, explaining why the variable total was local (question 8(c)(i)). However, explaining why the use of local variables makes a program more efficient proved to be more challenging (question 8(c)(ii)). Only a few candidates gained the mark, as responses often discussed modularity or maintainability rather than efficiency (question 8(c)(ii)).

Record data structures continue to be an area of strength for candidates. Most candidates showed an understanding of how records can store related data and could define a record structure and correctly declare an array of records using that structure (question 10(a)).

Candidates' understanding of procedural code was mixed. Many candidates provided the correct output from procedural code using the sample data provided, and explained why the procedure was not fit purpose. However, some candidates described a change to the stated purpose of the code rather than a change to the code itself (questions 9(a)(i), (ii) and (iii)).

In the linear search required for question 10(c), many candidates designed a suitable linear search and gained 3 or more marks. Common issues included incorrectly initialising the foundPosition and firstYear variables, which could result in incorrect output for some data sets.

Section 2: Database design and development

Overall performance in database design and development was stronger than in previous years.

Analysis and design

When analysing the problem to create functional requirements for a database, many candidates did not structure their responses appropriately (question 15(a)). A common weakness was not using phrasing such as ‘a query to...’, which would have helped candidates express the required database functionality and access both marks. Some candidates also simply restated the end-user requirements given in the stem.

Entity-relationship modelling continues to be a particular area of strength. Most candidates correctly interpreted the information provided and created an entity-relationship diagram showing the cardinality of the relationships between entities (question 11).

Implementation

Candidate performance in SQL-related questions was relatively strong. Many candidates interpreted an SQL query containing an aggregate function and WHERE, GROUP BY and ORDER BY clauses to work out the expected sorted output (question 12). Most candidates also successfully constructed SQL queries and demonstrated that they could create queries that used previously saved queries (question 14), and used the appropriate aggregate function with a GROUP BY clause (question 15(c)).

Section 3: Web design and development

Overall performance in web design and development was stronger than in previous years.

Analysis and design

Web design was a particular area of strength. Most candidates interpreted and evaluated a given navigational structure for a website against end-user requirements (question 16(b)). In addition, most candidates produced a suitable wireframe design that indicated validation, using the information provided (question 17). However, when analysing the problem to create functional requirements, most candidates did not make suitable reference to an HTML form or the use of JavaScript functionality (question 16(a)).

JavaScript and CSS implementation

Most candidates demonstrated a good understanding of event-driven functionality and created a suitable JavaScript function (question 18(a)(i)). Many candidates then correctly rewrote the HTML code, using the onmouseover and onmouseout events to call JavaScript functions (question 18(a)(ii)).

Candidates performed well on CSS-related tasks, with most using grouping selectors to make CSS rules more efficient (question 18(c)). While many candidates used the margin and padding attributes appropriately, some candidates confused the two (question 19(a)).

Assignment

Software design and development

Analysis, design and evaluation

While candidates generally engaged successfully with tasks, analysis, design and evaluation were the most challenging areas. Many candidates successfully identified most of the missing functional requirements to achieve 2 or 3 marks. However, some responses lacked an indication of the inputs coming from the file (task 1a).

Most candidates refined step 3 of the algorithm to gain at least 1 mark. However, some candidates did not include the necessary detail in their design, omitting one or more criteria or failing to return the fee array (task 1b).

Many candidates evaluated the usability of the program when used with a manufacturer name that was not in the file. However, some candidates made a statement of fact rather than providing an evaluative comment (task 1e).

Implementation and testing

Candidates continue to demonstrate strong programming skills. Most candidates implemented a modular program with appropriate sub-programs and standard algorithms. Candidates generally followed the supplied program data flow and design, but some did not include sufficient internal commentary to achieve the modular and maintainable mark (task 1c).

Performance in the testing task was also strong. Most candidates followed the design and applied the supplied test data to accurately determine the values after each iteration (task 1d(i)).

Database design and development

Overall performance in database design and development was stronger than in previous years.

Analysis and evaluation

While overall performance was good, the analysis and design tasks were the most challenging for candidates. Despite this, when creating two additional functional requirements, responses were generally well-structured, using the phrasing ‘a query to…’, which enabled many candidates to gain both marks (task 2a).

When evaluating the end-user requirements, many candidates explained why a specific end-user requirement could not be met by the existing database structure (task 2e).

Design, implementation and testing

Both of the design and implementation tasks were an area of strength for candidates.

Most candidates demonstrated a good understanding of entity-occurrence diagrams and added the correct number of instances and associations. However, a few

candidates added an extra instance of ‘genre’ that did not have a corresponding association (task 2b).

When implementing SQL statements, most candidates used aggregate functions appropriately and successfully incorporated the results of a saved query within a subsequent query to produce the correct output. Some candidates successfully implemented an alternative valid approach using a subquery (task 2c). Most candidates also constructed and implemented an SQL statement that joined multiple tables, and sorted data correctly to produce the required output (task 2d).

Candidates also performed well on the task requiring them to debug and amend a given SQL statement. Most candidates correctly identified both errors in the SQL statement and corrected them by adapting the criterion for released and adding the missing GROUP BY clause (task 2e).

Web design and development

Overall performance in web design and development was stronger than in previous years.

Design and implementation

Most candidates demonstrated a good understanding of web design and user navigation by successfully designing a multi-level website that met the requirements of the scenario. However, a few candidates neglected to indicate the navigation bar (task 3a).

Candidates were proficient in event-driven functionality and most implemented the desired outcome using JavaScript. Candidates used a variety of different approaches to achieve this successfully (task 3b).

Most candidates used CSS styling to control appearance and positioning, and implemented internal or external styling to achieve the design depicted by the wireframe. However, a few candidates did not implement a 15px gap between the categories, with some adding 15px to both sides, resulting in a 30px gap (task 3c).

Implementation of HTML forms was another aspect in which candidates demonstrated confidence, with most implementing forms with appropriate elements and validation to match the given design (task 3d(i)).

Testing and evaluation

While they implemented the forms well, most candidates did not describe how to fully test the select form element they had just implemented. Answers tended to lack the necessary detail to achieve the mark (task 3d(ii)).

When evaluating the website against the provided end-user requirements and required functionality, most candidates gave two reasons why the website was not fit for purpose (task 3e).

Preparing candidates for assessment

Question paper

Data representation, security risks, precautions and environmental impact

Candidates should ensure that they give specific answers to questions, such as explaining the advantages of cache memory or the accuracy of floating-point numbers, instead of generic facts.

You should consider further reinforcement of the different purposes of digital certificates and digital signatures. Digital signatures are created using the sender's private key and verify the integrity of a signed message through hashing and the use of the sender's public key. The digital certificate verifies the identity of the public key owner and confirms ownership of that key.

When answering environmental implication questions, candidates should ensure that they refer to relevant technology, such as sensors or GPS data.

Design and implementation

When completing data flow questions, candidates should ensure that they pass only the required parameters and use the data structures specified in the question.

You should expose candidates to as wide a variety of standard algorithms as possible to help them adapt these algorithms to an unfamiliar context.

Database design and development

When giving any functional requirements for a database, encourage candidates to start their answer with 'A query to...', as there are no Higher-level functional requirements which can be implemented without the use of an SQL query. Candidates should also be encouraged to identify the aggregate function(s) required for that query.

Web design and development

When stating functional requirements for web design, candidates should ensure that they reference appropriate Higher-level HTML content, such as HTML forms, or JavaScript code with suitable events.

Assignment

Once candidates complete their assignment, you should ensure that only pages containing candidate evidence are submitted and that these are presented in the correct order. Any pages containing only instructions must be discarded. Evidence should be clearly legible, particularly when screenshots of code are submitted.

Software design and development

While the practical implementation of code in the assignment remains an area of strength, candidates should ensure that their code is both modular and maintainable by using appropriate internal commentary, along with meaningful variable and sub-program names, whitespace, and indentation.

You should continue to provide candidates with opportunities to consolidate their skills in writing evaluative statements. These evaluations should refer to the candidate's own code and address the specific aspect of the code required by the question. Candidates should avoid generic statements or statements of fact that do not include evaluative judgement.

Database design and development

Similar to the question paper, when stating database functional requirements, candidates' answers should begin with 'A query to...', or similar. Encourage candidates to identify the aggregate function(s) or Higher-level concepts required to implement that query.

Web design and development

In web design and development, you should encourage candidates to check their output carefully to ensure that it matches the given requirements, rather than simply producing an output. This is particularly important when candidates use margins, as differences in spacing between elements can be subtle on screen.