Past-paper walk-through

Data Integrity ▪ 6.2

eXercise

Questions in this set

- 9618/12 N25 Q1 ▪ 2 marks
- 9618/21 N25 Q6 ▪ 9 marks
- 9618/11 J24 Q5 ▪ 16 marks
- 9618/11 J23 Q2 ▪ 16 marks

Reading the mark scheme

- **B1** a mark that stands on its own – the point is either made or not.
- **M1** a *method* mark: the right approach, even if the number is wrong.
- **A1** an *answer* mark, and it usually needs its M mark first.
- **C1** a working mark on the way to the answer.
- // separates answers the examiner accepts equally.
- **ecf** = error carried forward: a later part is marked on your own earlier answer.
- **owtte** = "or words to that effect" – the wording need not match.

Question 1

9618/12 N25 ▪ Q1 ▪ 2 marks

Draw one line from each verification method to indicate whether it is used during data transfer or data entry.

Verification Method

- Parity byte check
- Checksum
- Visual check
- Parity block check

Use

- Data transfer
- Data entry

[2]

Solution 1

Worked steps

Sort them by **what is being checked against what**.

- **Parity byte check** — data transfer. An extra bit per byte lets the receiver test what arrived against what was sent.
- **Checksum** — data transfer. A value calculated from the block is sent with it and recalculated on arrival.
- **Visual check** — data entry. A person compares what is on the screen against the source document; nothing is transmitted.
- **Parity block check** — data transfer. Parity applied across a whole block, so a single wrong bit can be located.

Solution 1

The rule that decides every row: verification during **transfer** is done by the machine comparing a computed value; verification during **entry** is done by a human (or by typing it twice) against the original.

Mark scheme

- 1 mark for 2 or 3 correctly connected verification method boxes, 2 marks for all 4 correct

Question 6

9618/21 N25 ▪ Q6 ▪ 9 marks

Students are learning about a simple check digit method for data validation. In this method, a single check digit is appended to the end of an original number to give a new number.
The students are studying a method which:

- calculates the sum of all the digits in the original number
- uses integer division to calculate the remainder when the sum is divided by 10
- uses the remainder as the check digit
- appends the check digit to the original number, creating the new number.

Question 6

For example:

original number	4162
sum of all digits	$4 + 1 + 6 + 2 = 13$
remainder when the sum is divided by 10 using integer division	3
new number	41623

Question 6

The method described can be used to detect single-digit errors. For example, if the new number is incorrectly input as 41633, then the check digit does not match and the input will be rejected.

(a) An incorrect attempt was made to enter 41623. The **first two** digits were entered incorrectly; the **last three** digits were entered correctly.

When this number was tested, the check digit was found to be correct and the input was accepted.

Identify an example of the incorrect attempt to enter 41623 and explain why this number would **not** be rejected. **[2]**

(b) A module `Generate()` will take an integer value representing an original number and return an integer value representing a new number which includes the check digit. The original number is always at least three digits in length.

Write pseudocode for the module `Generate()`

Question 6

Assume that the parameter is valid. **[7]**

Question 6

original number	4162
sum of all digits	4 + 1 + 6 + 2 = 13
remainder when the sum is divided by 10 using integer division	3
new number	41623

Solution 6

(a)

Mark scheme

- Number: Any number where the first two digits sum to 5 / 15 **and** the last three digits are 623
- Explanation: The remainder is the same / 3 //
- The sum of the (first four) digits is the same / 13

Solution 6

(b) Worked steps

Work out the shape before writing a line: sum the digits, take the remainder, stick it on the end.
Getting at the individual digits is the only real decision. Two ways work, and either scores: convert to a string and read one character at a time, or use MOD and DIV arithmetic.

```
FUNCTION Generate(Number : INTEGER) RETURNS INTEGER
    DECLARE Total, Index, CheckDigit : INTEGER
    DECLARE NumString : STRING

    Total <- 0
    NumString <- NUM_TO_STR(Number)
    FOR Index <- 1 TO LENGTH(NumString)
        Total <- Total + STR_TO_NUM(MID(NumString, Index, 1))
    NEXT Index

    CheckDigit <- Total MOD 10
    RETURN Number * 10 + CheckDigit
ENDFUNCTION
```

Solution 6

Four things earn the marks, and each is easy to drop:

- the **FUNCTION header with a typed parameter and a RETURNS type**, and a matching ENDFUNCTION;
- `Total` initialised to 0 **before** the loop;
- a loop that visits **every** digit — `LENGTH(NumString)`, not a fixed count, because the number's length varies;
- appending by `Number * 10 + CheckDigit`, which shifts the original left one place. Writing `Number + CheckDigit` adds it instead of appending, and loses the last mark.

Mark scheme

Example solution – conversion of Number to a string

Solution 6

FUNCTION Generate(Number : INTEGER) RETURNS INTEGER DECLARE Total, Index, CheckDigit : INTEGER DECLARE NumString : STRING
Total <- 0 NumString <- NUM$_T$O$_S$TR(*Number*)
FOR Index <- 1 TO LENGTH(NumString) Total <- Total +
STR$_T$O$_N$UM(*MID*(*NumString*, *Index*, 1))*NEXTIndex*
CheckDigit <- Total MOD 10 Number <- Number * 10 + CheckDigit // NumString
<- Numstring
NUM$_T$O$_S$TR(*CheckDigit*)*RETURNNumber*//STR$_T$O$_N$UM(*NumString*)*ENDFUNCTION*
Mark as follows:

Solution 6

1. Function heading, parameters return type and ending
2. Initialise `Total`
3. Convert `Number` to a string and loop for length of converted `Number`
4. attempt to form sum of digits **in a loop**
5. completely correct sum of digits **in a loop**
6. Calculate `CheckDigit`
7. Add `CheckDigit` to original number multiplied by 10 and return number
   //
   Convert `CheckDigit` to a character and appended to `NumString` and then convert to a number and return

Example Solution – no conversion to string

Solution 6

FUNCTION Generate(Number : INTEGER) RETURNS INTEGER DECLARE Total, CheckDigit, Remainder, Num: INTEGER
Num <- Number Total <- 0
WHILE Num > 0 Remainder <- Num MOD 10 Num <- Num DIV 10 Total <- Total + Remainder ENDWHILE
CheckDigit <- Total MOD 10 Number <- Number * 10 + CheckDigit
RETURN Number ENDFUNCTION
Mark as follows:

Solution 6

1. Function heading, parameters return type and ending
2. Initialise `Total`
3. Loop while `Num` $>$ 0
4. attempt to use MOD and DIV to sum successive digits in `Number`
5. completely correct sum of digits
6. Calculate `CheckDigit`
7. Add `CheckDigit` to original number **and** return number

Question 5

9618/11 J24 ▪ Q5 ▪ 16 marks

5 A bank allows customers to access their accounts using an application that they can download onto a device such as a smartphone.

(a) The system that allows customers to access their accounts using the application is a client-server model.

Describe the roles of the different devices in this model.

Question 5

(b) The bank wants to protect the integrity of its data while transferring the data to other banks. Parity check is one example of data verification.

Complete the description of parity check when Computer A is transmitting data to Computer B.

Question 5

and the group has an even number of 1s, a parity bit of 1 is appended, otherwise a parity bit of 0 is appended.

In a parity ..... check the bytes are grouped together, for example in a grid. The number of 1s in each column (bit position) is counted. A bit is assigned to each column to make the column match the parity. These parity bits are transmitted with

Question 5

[5]

(c) The bank also needs to keep its customers' data private and secure.

(i) The bank's network has a firewall.

Explain how a firewall can help protect the customers' data.

Question 5

(ii) Customers need to use biometric authentication to access their accounts.
One biometric authentication method is facial recognition.

Facial recognition uses Artificial Intelligence (AI).

Describe how AI is used in facial recognition.

Solution 5

(a) Mark scheme

- 1 mark each:
- ▪
- Identification of server in the bank scenario
- ▪
- Description e.g. Receives requests, processes the requests
- ▪
- Identification of client in bank scenario
- ▪

Solution 5

- Description e.g. Sends request to the server, waits and outputs the
- response
- 4

Solution 5

(b) Mark scheme

- 1 mark for each correctly completed term:
- ▪
- odd or even
- ▪
- 7-bits
- ▪
- odd
- ▪

Solution 5

- block
- ▪
- byte
- Computer A and Computer B agree on whether to use odd or even parity.
- Computer A divides the data into groups of 7-bits. The number of 1s in each
- group is counted. If the agreed parity is odd and the group has an even
- number of 1s a parity bit of 1 is appended, otherwise a parity bit of 0 is
- appended.
- In a parity block check the bytes are grouped together, for example in a grid.
- The number of 1s in each column (bit position) is counted. A bit is assigned to

Solution 5

- each column to make the column match the parity. These parity bits are
- transmitted with the data as a parity byte.
- 5

Solution 5

(c)(i) Worked steps

Three marks, three steps of the same mechanism:

- It **inspects every transmission** entering or leaving the system.
- It compares each against **set criteria** — a whitelist of what is allowed, or a blacklist of what is not.
- It **blocks** anything that fails those criteria, and can log the attempt.

Saying a firewall "stops hackers" scores nothing; the marks are for inspect, compare, block. **Mark scheme**

- 1 mark each to max 3:

Solution 5

- ▪
- Compares all incoming and outgoing transmissions
- ▪
- ... against set criteria/whitelist/blacklist
- ▪
- Blocks all transmissions that do not meet rules
- ▪
- Blocks data entering from specific ports
- ▪
- Blocks unauthorised/unknown internal software transmitting data

Solution 5

- 3

Solution 5

(c)(ii) Worked steps

- The system **captures an image** of the face with a camera.
- It uses **image recognition** to locate the face and measure its features — the distances between eyes, nose and mouth, for example.
- Those measurements are compared against the **stored template** for that user.
- If they match within a set tolerance the user is authenticated; otherwise access is refused.

The tolerance matters and is worth stating: a face is never captured identically twice, so biometric matching is a similarity threshold rather than an exact comparison — unlike a password. **Mark scheme**

Solution 5

- 1 mark each to max 4:
- e.g.
- ▪
- Captures an image of the face
- ▪
- Uses image recognition
- ▪
- Trained to identify the features of a face in an image
- ▪
- ... using a large number of images

Solution 5

- ▪
- Analyse images for facial features
- ▪
- Uses the probability of a match
- 4
- 9618/11
- May/June 2024

Question 2

9618/11 J23 ▪ Q2 ▪ 16 marks

An organisation uses a database to store data about the types of bird that people have seen.

(a)(i) State what is meant by a data dictionary and give **one** example of an item typically found in a data dictionary. **[2]**

(a)(ii) State what is meant by data integrity and give **one** example of how this is implemented in a database. **[2]**

Question 2

Foreign key	Database table

[2]

Question 2

BirdID	Name	Size
0123	Blackbird	Medium
0035	Jay	Large
0004	Raven	Large
0085	Robin	Small

Question 2

9618/11 J23 ▪ Q2 ▪ 16 marks

An organisation uses a database to store data about the types of bird that people have seen.

(b)(i) Complete the table by identifying **two** foreign keys and the database table where each is found.

Foreign key	**Database table**

Question 2

[2]

(b)(ii) The database `Birds` has been normalised.
Draw one line from each Normal Form to the most appropriate definition.

Normal Form	**Definition**
First Normal Form (1NF)	All fields are fully dependent on the primary key.
Second Normal Form (2NF)	There are no repeating groups of attributes.
Third Normal Form (3NF)	There are no partial dependencies.

Question 2

[1]

(b)(iii) Part of the database table `BIRD_TYPE` is shown:

BirdID	Name	Size
0123	Blackbird	Medium
0035	Jay	Large
0004	Raven	Large
0085	Robin	Small

Question 2

The database only supports these data types:

- character
- varchar
- Boolean
- integer
- real
- date
- time

Question 2

Write a Structured Query Language (SQL) script to define the table `Bird_Type`. **[4]**

(b)(iv) The database tables are repeated here for reference:

$BIRD_TYPE(BirdID, Name, Size)$ $BIRD_SEEN(SeenID, BirdID, Date, Location, PersonID)$ $PERSO$

Complete the SQL script to return the number of birds of each size seen by the person with the ID of J_123.

SELECT

$BIRD_TYPE.Size,$ _____ $(BIRD_TYPE.BirdID)$ $ASNumberOfBirdsFROMBIRD_TYPE$

$"J_123"ANDBIRD_TYPE.BirdID =$

_____ $BIRD_TYPE.Size;$ **[5]**

Solution 2

(a)(i)

Mark scheme

1 mark for definition:

- Data about the data in the database // data about the structure of the database // metadata for a database

1 mark for a suitable example

Examples:

- table names
- data types
- field names

Solution 2

(a)(ii)

Mark scheme

1 mark for definition

- Methods of making sure the data is consistent

1 mark for example

Examples:

- Enforcing referential integrity
- If data in one table is deleted/edited all tables are updated // cascading update/delete
- Validation/verification rules

Solution 2

(b)(i) Mark scheme

1 mark for each field name and table

Foreign key	Database table
BirdID	BIRD_SEEN
PersonID	BIRD_SEEN

Solution 2

(b)(ii) Worked steps

Know. The three normal forms are a ladder. Each one assumes the previous one is already done, and each removes one specific kind of repetition.
Match them.

- **1NF** connects to *"There are no repeating groups of attributes."* First you flatten the table so no field holds a list.
- **2NF** connects to *"There are no partial dependencies."* Then you remove any field that depends on only **part** of a composite primary key.
- **3NF** connects to *"All fields are fully dependent on the primary key."* Finally you remove fields that depend on another **non-key** field rather than on the key itself.

Solution 2

Check. All three lines have to be right for the single mark, so match the two you are sure of first and let the third fall out. The wording is the giveaway: "repeating groups" is 1NF's own phrase, "partial" points at part of a key, which needs a composite key and so belongs to 2NF, and the survivor is 3NF.

Remember it as. Every non-key field depends on **the key** (1NF), **the whole key** (2NF), and **nothing but the key** (3NF).

Mark scheme

1 mark for all 3 correct lines

- First Normal Form (1NF) connects to: There are no repeating groups of attributes.
- Second Normal Form (2NF) connects to: There are no partial dependencies.
- Third Normal Form (3NF) connects to: All fields are fully dependent on the primary key.

Solution 2

(b)(iii)

Mark scheme

1 mark each

- CREATE TABLE start and end bracket
- Bird ID as CHAR/VARCHAR
- Name and size as VARCHAR/CHAR
- Bird ID as primary key

Example answer: CREATE TABLE BIRD$_TYPE$(*BirdIDCHAR*(4)*NOTNULL*, *NameVARCHA*

Solution 2

(b)(iv) Mark scheme

- **1 mark for each correctly completed space**
-
- SELECT BIRD_TYPE.Size, COUNT(BIRD_TYPE.BirdID) AS NumberOfBirds
- FROM BIRD_TYPE, BIRD_SEEN
- WHERE BIRD_SEEN.PersonID = "J_123"
- AND BIRD_TYPE.BirdID = BIRD_SEEN.BirdID
- GROUP BY BIRD_TYPE.Size;
-