ANSWERS

A11.1 Data models and entity relationships

Databases and big data · A level · OCR H446 1.3.2, AQA 7517 4.10.1, Eduqas A500QS 2.5 · about 55 min

What this lesson is about

Model the data before building anything: entities, attributes, entity descriptions and the degree of each relationship, drawn as an entity relationship diagram.

- Conceptual data models
- Entity descriptions
- The degree of a relationship
- Resolving a many-to-many relationship
- Checking a degree against data

Questions

6 marks in all

1. In a data model, what is an entity? [1 mark]

- ☐ **A** A category of object, person, event or thing about which data is recorded
- ☐ **B** A single property, such as a robot's colour
- ☐ **C** A line joining two tables
- ☐ **D** One row of a table

> **Answer:** A. An entity is a thing of interest, such as Robot; a property of it is an attribute.

2. A team owns many robots, and each robot belongs to exactly one team. What is the degree of the relationship from Team to Robot? [1 mark]

- ☐ **A** One-to-many
- ☐ **B** Many-to-one
- ☐ **C** One-to-one
- ☐ **D** Many-to-many

> **Answer:** A. One team is linked to many robots, and each robot to one team.

3. In an E-R diagram of that relationship, where is the crow's foot drawn? [1 mark]

- ☐ **A** At the Robot end
- ☐ **B** At the Team end
- ☐ **C** At both ends
- ☐ **D** At neither end

> **Answer:** A. The crow's foot goes at the many end, and there are many robots per team.

4. How is a many-to-many relationship between Student and Robot implemented in a relational database? [1 mark]

- ☐ **A** With a link entity, such as Booking, holding both keys, giving two one-to-many relationships
- ☐ **B** By storing a list of robot IDs in one Student attribute
- ☐ **C** By merging Student and Robot into one table
- ☐ **D** It cannot be stored at all

> **Answer:** A. A link entity turns one many-to-many relationship into two one-to-many relationships; an attribute cannot hold a list.

5. In the entity description Robot (RobotID, Name, Colour, TeamID), which attribute should be underlined? [1 mark]

> **Answer:** RobotID. The entity identifier is underlined, and RobotID is unique for every robot.

6. A sample of bookings shows no student has booked more than one robot. What can you conclude? [1 mark]

- ☐ **A** Nothing certain: a sample can show a side is many, but only the requirements can say it is one
- ☐ **B** The relationship is one-to-one
- ☐ **C** The relationship is one-to-many
- ☐ **D** Students cannot book robots

> **Answer:** A. Data can disprove a degree of one but never prove it; the degree comes from the requirements.

The task: the degree of a relationship

Each list below holds pairs `(left, right)` taken from the club's records. Write a function `degree(pairs)` that takes one list of 2-tuples and returns a string: `"one-to-one"`, `"one-to-many"`, `"many-to-one"` or `"many-to-many"`. - The **right** side is `many` if any left value appears with two or more different right values; otherwise it is `one`. - The **left** side is `many` if any right value appears with two or more different left values; otherwise it is `one`. - The string is the left side, then `-to-`, then the right side. Call it on each list and print four lines, in this order, exactly like this (with the degree your function returns): The robot does not move.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

teams_robots = [("Hawks", "Ada"), ("Hawks", "Bolt"), ("Owls", "Cog"), ("Owls", "Dot")]
students_robots = [("Amir", "Ada"), ("Beth", "Ada"), ("Amir", "Cog"), ("Chen", "Bolt"),
("Beth", "Dot")]
robots_chargers = [("Ada", "C1"), ("Bolt", "C2"), ("Cog", "C3"), ("Dot", "C4")]
runs_robots = [(1, "Ada"), (2, "Ada"), (3, "Bolt"), (4, "Cog")]
```

The hint students can ask for: For each left value, collect the set of right values it is paired with, and the other way round. If any left value has more than one partner, the right-hand side is many. Then do the same for the right values.

A solution

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

teams_robots = [("Hawks", "Ada"), ("Hawks", "Bolt"), ("Owls", "Cog"), ("Owls", "Dot")]
students_robots = [("Amir", "Ada"), ("Beth", "Ada"), ("Amir", "Cog"), ("Chen", "Bolt"),
("Beth", "Dot")]
robots_chargers = [("Ada", "C1"), ("Bolt", "C2"), ("Cog", "C3"), ("Dot", "C4")]
runs_robots = [(1, "Ada"), (2, "Ada"), (3, "Bolt"), (4, "Cog")]

def degree(pairs):
    lefts = {}
    rights = {}
    for a, b in pairs:
        lefts.setdefault(a, set()).add(b)
        rights.setdefault(b, set()).add(a)
    left_side = "many" if any(len(s) > 1 for s in rights.values()) else "one"
    right_side = "many" if any(len(s) > 1 for s in lefts.values()) else "one"
    return left_side + "-to-" + right_side

print("teams to robots:", degree(teams_robots))
print("students to robots:", degree(students_robots))
print("robots to chargers:", degree(robots_chargers))
print("runs to robots:", degree(runs_robots))
```

Any program that meets the task's checks is marked correct in the simulator; this is one way, not the only way.