ANSWERS

F7.2 Relational databases

Files and databases · GCSE · OCR J277 2.2.3, AQA 8525 3.7.1 · about 15 min

What this lesson is about

Tables, records, fields, primary and foreign keys, and avoiding redundancy.

- Tables, records and fields
- The trouble with one big table
- Splitting into linked tables
- Keys
- A database in Python
- Following a foreign key

Questions

6 marks in all

1. In a database table, what is a record? [1 mark]

- ☐ **A** One row: all the data about one item
- ☐ **B** One column
- ☐ **C** The whole table
- ☐ **D** The primary key

> **Answer:** A. Records are rows; fields are columns.

2. What is a primary key? [1 mark]

- ☐ **A** A field whose value is unique for every record in the table
- ☐ **B** The first field in any table
- ☐ **C** A password for the database
- ☐ **D** A field that links to another table

> **Answer:** A. It identifies each record exactly. A field that links to another table is a foreign key.

3. In the runs table, robot_id holds the primary key of a record in the robots table. What is robot_id in runs? [1 mark]

- ☐ **A** A foreign key
- ☐ **B** A primary key
- ☐ **C** A record
- ☐ **D** A table

> **Answer:** A. A foreign key links a record to a record in another table.

4. Ada's colour is typed into every one of her runs. What is this called? [1 mark]

- ☐ **A** Data redundancy
- ☐ **B** Data validation
- ☐ **C** A primary key
- ☐ **D** Normal data

> **Answer:** A. The same data is stored more than once.

5. Why is storing Ada's colour once, in a robots table, better? [1 mark]

- ☐ **A** It cannot become inconsistent: changing it once changes it everywhere
- ☐ **B** It makes the database slower
- ☐ **C** It removes the need for keys
- ☐ **D** It hides the colour

> **Answer:** A. Linked tables reduce redundancy and the inconsistency it causes.

6. Why would a robot's name make a poor primary key? [1 mark]

- ☐ **A** Two robots could have the same name
- ☐ **B** Names are too long
- ☐ **C** Names cannot be stored in databases
- ☐ **D** Names are numbers

> **Answer:** A. A primary key must be unique for every record.

The task: runs by robot

Using `logbook.open_db()`, print one line for every run, in run order, in the form `run 1 by Ada: wall 42.0 cm`. Look each robot's name up through the `robot_id` foreign key; do not type the names into your program.

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

import logbook

db = logbook.open_db()
```

The hint students can ask for: Build a dictionary from robot_id to name from SELECT * FROM robots, then loop over SELECT * FROM runs and look each robot_id up.

A solution


```
{'program': 'from bugbot import *\nconnect()\nimport logbook\n\ndb =
logbook.open_db()\nnames = {}\nfor robot_id, name, colour in db.execute("SELECT * FROM
robots"):\n    names[robot_id] = name\nfor run_id, robot_id, task, cm, seconds in
db.execute("SELECT * FROM runs ORDER BY run_id"):\n    print(f"run {run_id} by
{names[robot_id]}: {task} {cm} cm")\n', 'files': {'logbook.py': '# logbook.py: the class\'s
robot runs as a relational database\nimport sqlite3\n\nROBOTS = [(1, "Ada", "green"), (2,
"Bolt", "red"), (3, "Cog", "blue")]\nRUNS = [(1, 1, "wall", 42.0, 3.1), (2, 1, "square",
80.0, 9.4), (3, 2, "wall", 38.0, 2.7),\n        (4, 1, "wall", 45.5, 3.0), (5, 3, "square",
76.5, 8.8), (6, 2, "square", 81.0, 10.2)]\n\ndef open_db():\n    """A fresh database with
the robots and runs tables."""\n    db = sqlite3.connect(":memory:")\n
db.execute("CREATE TABLE robots (robot_id INTEGER PRIMARY KEY, name TEXT, colour TEXT)")\n
db.execute("CREATE TABLE runs (run_id INTEGER PRIMARY KEY, robot_id INTEGER REFERENCES
robots(robot_id), "\n                "task TEXT, cm REAL, seconds REAL)")\n
db.executemany("INSERT INTO robots VALUES (?, ?, ?)", ROBOTS)\n    db.executemany("INSERT
INTO runs VALUES (?, ?, ?, ?, ?)", RUNS)\n    return db\n\ndef show(db, sql):\n    """Run a
query and print each record on its own line."""\n    for row in db.execute(sql):\n
print(*row)\n'}}
```


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