



A11.3 Normalisation to third normal form

Databases and big data · A level · OCR H446 1.3.2, AQA 7517 4.10.3,
Eduqas A500QS 2.5 · about 60 min

Name _____

Class _____

Date _____

What this lesson is about

Functional dependencies, update anomalies, and taking a flat score sheet through first, second and third normal form.

- What goes wrong without it
- Functional dependency
- Unnormalised form
- First normal form
- Second normal form
- Third normal form
- Why normalise, and when not to

Questions

6 marks in all

1. A table is in 1NF and its primary key is a single attribute. Which normal form must it also be in? [1 mark]
☐ A 2NF
☐ B 3NF
☐ C Only 1NF
☐ D None
2. Robot (RobotID, Name, TeamID, TeamName). TeamName depends on TeamID. Which rule does the table break? [1 mark]
☐ A 3NF: a non-key attribute depends on another non-key attribute
☐ B 1NF: it has a repeating group
☐ C 2NF: an attribute depends on part of a composite key
☐ D It breaks no rule
3. Score (RobotID, TaskCode, TaskName, Score) has the key (RobotID, TaskCode). Why is it not in 2NF? [1 mark]
☐ A TaskName depends on TaskCode, only part of the key
☐ B Score depends on the whole key
☐ C It has no primary key
☐ D RobotID is a foreign key
4. A team has no robots yet, and a flat score table cannot record the team without a fake score. What is this called? [1 mark]
☐ A An insertion anomaly
☐ B An update anomaly
☐ C A deletion anomaly
☐ D A referential integrity error

5. Put the steps of normalisation in order.

[1 mark]

Number the lines 1 to 3 to put them in the right order.

- ___ Remove partial dependencies on part of a composite key (2NF)
- ___ Remove dependencies between non-key attributes (3NF)
- ___ Remove repeating groups and choose a primary key (1NF)

6. What does this print?

[1 mark]

```
rows = [("T1", "Hawks"), ("T1", "Hawks"), ("T2", "Owls"), ("T1", "Hawks")]
teams = sorted(set(rows))
print(len(rows), "records became", len(teams))
print(teams)
```

The task: normalise the score sheet

The list `flat` is the 1NF score sheet: each item is a tuple (`robot_id`, `robot_name`, `team_id`, `team_name`, `task_code`, `task_name`, `score`), where `score` is a whole number and the rest are strings. Build the four 3NF tables from `flat` (you choose the Python structure), with each fact stored once, and print them in this order, each table sorted by its key: - teams, as `team <team_id> <team_name>` - robots, as `robot <robot_id> <robot_name> <team_id>` - tasks, as `task <task_code> <task_name>` - scores, sorted by robot then task, as `score <robot_id> <task_code> <score>` That is 15 lines, starting `team T1 Hawks` and ending `score R4 L 60`. The robot does not move.

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

# robot_id, robot_name, team_id, team_name, task_code, task_name, score
flat = [
    ("R1", "Ada", "T1", "Hawks", "W", "wall", 42),
    ("R1", "Ada", "T1", "Hawks", "S", "square", 80),
    ("R2", "Bolt", "T1", "Hawks", "W", "wall", 38),
    ("R3", "Cog", "T2", "Owls", "S", "square", 76),
    ("R3", "Cog", "T2", "Owls", "W", "wall", 45),
    ("R4", "Dot", "T2", "Owls", "L", "line", 60),
]
```

Plan your program here, then type it in and press Run.



Do it on the robot

www.bugbotlab.com/learn/a11-3-normalisation/

The simulator checks it and tells you when it passes. Nothing to install, no account.

Challenges

1. A library loan table is `Loan (<u>BookID, MemberID, LoanDate</u>, Title, Author, MemberName, MemberPostcode, ReturnDate)`. Normalise it to 3NF.
2. Add a `Coach` attribute that depends on the team. Where does it go in 3NF?
3. Use `determines` to test `Score` against each of the other attributes on its own. Why does none of them determine it?