A3 DATABASES · A3.2

Database design

Turning a problem into clean tables: the **schema**, **ERDs**, data types, and **normalisation** to third normal form (with a full worked example on page 3).

01 Schema & ERD

Schema	The structure: entities, types, keys.
Entity	A table.
Attribute	A field, with a data type.
Relationship	A link between entities.
Cardinality	1:1, 1:N, M:N (crow's foot = many).
ERD	A visual model of the schema.

02 Data types

INTEGER	Whole numbers (42).
REAL	Decimals (3.14).
CHAR(n)	Fixed-length text.
VARCHAR(n)	Variable-length text (a name).
BOOLEAN	True or false.
DATE/ TIME	Dates and times.

03 The normal forms

UNF	Unnormalised: the raw data, with repeating groups and non-atomic cells.	START
1NF	Every cell holds a single (atomic) value and there are no repeating groups.	ATOMIC
2NF	In 1NF, no partial dependencies: every non-key attribute depends on the whole (composite) key.	PARTIAL
3NF	In 2NF, no transitive dependencies: non-key attributes depend only on the key, not on each other.	TRANSITIVE

04 Keys & integrity

Primary key	Unique, never empty.
Foreign key	Holds another table's primary key.
Composite	Two or more fields as the key.
Referential	An FK must match an existing PK.
Atomic	One value per cell.

05 Denormalisation

What	Re-add some redundancy on purpose.
Why	Faster reads (reporting, analytics).
Cost	Harder to keep data consistent.
When	A performance need, not the default.

06 The recipe · what each step removes

1NF	Split repeating groups into their own rows/table so every cell is atomic. Give the new table a key (often composite).	REPEATING
2NF	Move attributes that depend on only part of a composite key into their own table (e.g. ModuleName depends on ModuleID, not the whole key).	PARTIAL DEP
3NF	Move attributes that depend on another non-key field (e.g. MentorName depends on MentorID) into their own table.	TRANSITIVE DEP
Check	Every non-key attribute now depends on "the key, the whole key, and nothing but the key".	DONE

07 Worked example · a coding academy, UNF → 3NF

UNF · raw data — the "Modules taken" cell is a repeating group

StudentID	StudentName	MentorID	MentorName	Modules taken (ID, Name, Level, Score)
S1	Aisha	M2	Ms Patel	P101 Python Basics (1) 78; P205 Web Dev (2) 65
S2	Ben	M1	Mr Ng	P101 Python Basics (1) 81

1NF · atomic cells; split the repeating group. New key = (StudentID, ModuleID)

STUDENT

StudentID (PK)	StudentName	MentorID	MentorName
S1	Aisha	M2	Ms Patel
S2	Ben	M1	Mr Ng

ENROLMENT · PK (StudentID, ModuleID)

StudentID	ModuleID	ModuleName	Level	Score
S1	P101	Python Basics	1	78
S1	P205	Web Dev	2	65
S2	P101	Python Basics	1	81

2NF · remove partial deps — ModuleName/Level depend only on ModuleID, so move them to MODULE

ENROLMENT

StudentID (PK,FK)	ModuleID (PK,FK)	Score
S1	P101	78
S1	P205	65
S2	P101	81

MODULE

ModuleID (PK)	ModuleName	Level
P101	Python Basics	1
P205	Web Dev	2

3NF · remove transitive dep — MentorName depends on MentorID (not the key), so move it to MENTOR

STUDENT

StudentID (PK)	StudentName	MentorID (FK)
S1	Aisha	M2
S2	Ben	M1

MENTOR

MentorID (PK)	MentorName
M1	Mr Ng
M2	Ms Patel

Result · four tables (STUDENT, MENTOR, MODULE, ENROLMENT), each fact stored once. Renaming a mentor is now one edit.

FINAL PASS BEFORE THE EXAM

Rapid exam tips

Eight things that lose marks in Paper 1 if you slip on them. Skim before you walk in.

01

1NF = atomic values, no repeating groups.

02

2NF = 1NF + no partial dependencies (only matters with a composite key).

03

3NF = 2NF + no transitive dependencies (non-key fields depend only on the key).

04

In 2NF, only move attributes that depend on **part** of the key; ones that depend on the **whole** key (like Score) stay.

05

Mark the **PK** (unique, not null) and **FK** (links to a PK) on every ERD.

06

ERD: entity = box, attribute = field, line = relationship. The **crow's foot** is the "many" end.

07

Denormalisation trades tidiness for read speed; it is a deliberate choice.

08

Referential integrity: a foreign key must match an existing primary key.