INFO90002 Database Systems and Data Modelling

SCHOOL OF COMPUTING & INFORMATION SYSTEMS · THE UNIVERSITY OF MELBOURNE

Full guide + ask Sia about INFO090002

SIDE 1/2 WHAT MUST BE IN YOUR HEAD · PRIORITY RULES + CHECKS **CLOSED-BOOK MEMORY SHEET** S2 2026 · postgraduate · v2

A1 20%	A2 10%	Q1 15%	Q2 5%	END-OF-SEMESTER EXAMINATION 50%

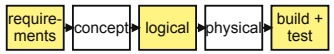
EXAM MAP · OFFICIAL ASSESSMENT CONTROL

Assignment 1: Data Modelling 20% · Assignment 2: SQL 10% · Quiz 1 15% · Quiz 2 5% · End-of-Semester Examination 50% · Coursework Hurdle 25/50 minimum · Examination Hurdle 25/50 minimum. "Quiz 1 and Exam are closed book, invigilated" — INFO090002 2026 SM2 Week 1 admin slides (PDF_INFO090002_Week1_1_AdminIntro_pdf.txt).

01 · Lifecycle = level first C1 · C2

RECITE BEFORE ANY DESIGN ANSWER

Requirements → conceptual → logical → physical → implementation → validation. Conceptual models business entities/relationships without technology; logical translates them into a data model + constraints; physical specifies storage, indexes and implementation details.



Lifecycle chain: every stage changes the artefact and the check. Source: assessment-evidence…fig1.

DECISION RULE

IF THE PROMPT ASKS…	ANSWER AT…
Business meaning	Conceptual: entities, attributes, relationships
Relations + constraints	Logical
Storage/index detail	Physical
Whether it works	Validation: normal, boundary, invalid

★ *Exam bite:* name the artefact at the current level; do not repair a missing business rule with implementation detail. Trace what reads, changes or depends on it.

02 · Requirement → model C2



Requirement chain: rule → identifiable thing → justified relationship. Source: requirements…fig1.

ENTITY TEST · WHERE IT BITES

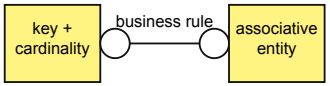
Promote a noun only when the business must identify instances and retain facts about them. Otherwise it may be an attribute, role or narrative wording.

- Extract declarative business rules; separate required facts from interface or process wording.
- Name candidate entities, identifiers, attributes and relationships; make each constraint testable.
- For every relationship, state the business sentence using **must** or **may**.
- Ambiguity remains a requirements question; notation cannot manufacture the missing rule.

ATTRIBUTE CLASSIFIER

03 · ER notation & constraints C3

TERM	EXACT JOB
Identifier	Uniquely distinguishes entity occurrences; becomes a primary key.
Connectivity	Relationship type, e.g. one-to-many.
Cardinality	Minimum + maximum participation counts.
Participation	Optional min 0; mandatory min 1.
Associative entity	Represents a relationship as an entity when the model requires it.



Constraint bridge: key/cardinality ↔ business rule ↔ associative entity. Source: er-diagrams…fig1.

NOTATION COMPARISON · EXAM USE

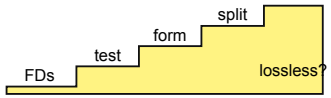
Crow's Foot: entity boxes + line-end cardinality symbols. **Chen:** rectangles (entities), ovals (attributes), diamonds (relationships). Notation records a rule; it does not resolve ambiguity.

Response order: identify occurrence → name identifier → state each min/max in words → justify with the business rule → use an associative entity only when the rule demands it.

04 · Normalisation method C4

CORE RULE · HIGHEST PRIORITY

X → Y is a rule about all valid tuples, not a coincidence in sample rows. Normalisation decomposes relations according to functional dependencies to reduce redundancy and prevent insertion, update and deletion anomalies while preserving required information.



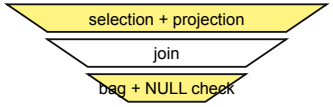
Normalisation staircase: dependencies → form test → decomposition → lossless check. Source: logical-design…fig1.

1. Write the relation and the claimed functional dependencies as business rules.
2. Use those dependencies—not repeated-looking values—to test the target normal form.
3. Decompose only to remove the dependency-driven anomaly.
4. Check **lossless decomposition**: required information and joinability survive.

05 · Relational operations first C5

PREDICT BEFORE RUNNING SQL

Selection restricts rows (tuples). **Projection** restricts columns (attributes). **Join** combines matching tuples. Then check SQL's duplicate and NULL behaviour explicitly.



Query-result funnel: restrict → combine → test SQL semantics. Source: relational-algebra…fig1.

QUESTION	OPERATOR/CHECK
Which rows?	Selection / WHERE
Which attributes?	Projection / SELECT list
Which relations combine?	Join + matching condition
Why extra rows?	Join multiplicity; bag duplicates
Why no match?	NULL can yield UNKNOWN

★ *Exam bite:* state the input relation(s), apply one operation at a time, preserve the intended row grain, and only then predict the output.

BOUNDARY *Pure relational intuition is set-based; SQL duplicates and NULL logic can change the observed result.*

06 · DDL, DML, DCL + integrity C6

CLASS	COMMANDS	PURPOSE
DDL	CREATE, ALTER, DROP	Define structures
DML	SELECT, INSERT, UPDATE, DELETE	Query/change data
DCL	GRANT, REVOKE	Manage privileges



Integrity matrix: structure + constraint governs every write. Source: sql-ddl-dml…fig1.

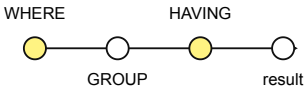
STRONGEST-BOUNDARY RULE

Encode domain, key and referential rules as database constraints when every writer must obey them. Application validation improves messages, but another client can bypass it.

07 · Grouping, subqueries, views C7

GRAIN RULE · WHERE MARKS DISAPPEAR

WHERE filters rows before grouping; HAVING filters completed groups. Nested queries and views are alternative query shapes; use them only when they preserve the intended row and group grain.



Aggregation order: row filter → form groups → group filter → result. Source: sql-aggregation…fig1.

TRUST-BUT-VERIFY CHECKLIST

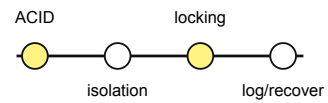
RISK	CHECK
Inflated aggregate	Inspect join cardinality before grouping
Wrong filter level	Row predicate → WHERE; group predicate → HAVING
Scalar subquery failure	Does it really return one value?
View hides mistake	Verify its row grain and constraints first

CLASSIC TRAP *A correct-looking scalar can still come from an accidentally duplicated join. Plausibility is not validation.*

08 · Transactions + recovery C8

ACID RECALL

A transaction is a logical unit of work that should satisfy **ACID**. Concurrency control and locking coordinate simultaneous transactions to prevent inconsistent or lost updates; logging supports recovery.



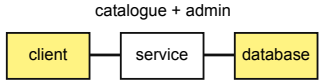
Transaction chain: required guarantee → isolation/control → recovery support. Source: transactions…fig1.

1. Trace the interleaving in order; identify the inconsistent or lost update.
2. Name the property/invariant at risk and the isolation behaviour needed.
3. Explain how locking or versioning enforces the choice.
4. Explain how logging/recovery handles failure.
5. Qualify: serialisable behaviour may add blocking, retries or deadlocks.

09 · Architecture + distribution C9

RESPONSIBILITY PLACEMENT

Three-tier architecture separates **client**, **service** and **database** responsibilities. Catalogue/administration supplies metadata and operational control. Distributed data adds node/link failure and consistency choices.



Three-tier responsibility path with distributed failure boundary. Source: architecture…fig1.

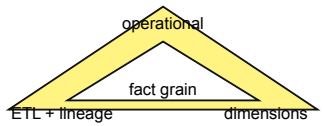
PROMPT CLUE	ASK
User interaction	What belongs at client?
Business orchestration	What belongs at service?
Integrity/persistence	What must database enforce?
Metadata/admin	Which catalogue fact/control?
Remote node/link	What fails; what consistency/recovery choice follows?

BOUNDARY *Distribution relocates consistency and recovery decisions; it does not make them disappear.*

10 · Warehouse reasoning C10

GRAIN BEFORE OPTIMISATION

Separate **operational** from **analytical** data. Declare the fact table's grain before assigning measures and dimension keys. ETL lineage preserves how every metric was derived from operational data.



Warehouse discriminator: source/lineage + fact grain + dimensions. Source: data-warehousing…fig1.

STAGE	REQUIRED STATEMENT
Classify	Operational or analytical purpose?
Grain	Exactly what one fact row represents
Model	Measures + dimension keys at that grain
ETL	Transformation from source to metric
Lineage	How definition and origin remain traceable

A warehouse metric needs a stable definition before optimisation. Performance cannot repair inconsistent meaning across loads.

11 · NoSQL / JSON chooser C11

MODEL-SELECTION RULE

Choose a document model from access pattern, integrity, update boundary, consistency and evolution needs—not from the word "NoSQL". Choose embedding or references by the same evidence.



Document choice is bounded by access, integrity, evolution and consistency. Source: nosql-json…fig1.

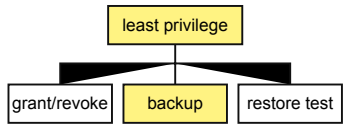
CLAIM	REQUIRED QUALIFICATION
"Flexible schema"	Field meaning/types still constrain use
"Schema-on-read"	Moves the enforcement boundary; does not abolish schema
"Embed"	Defend with access + update boundary
"Reference"	Defend with access pattern + consistency need

EXAM TEST *State the precondition for the document choice, reject the nearest alternative, then run the consistency failure path.*

12 · Security, backup, integration C12

TWO INDEPENDENT CONTROLS

Least privilege: grant only what the stated role needs. **Recoverability:** prove it with a restore test, not the existence of a backup file.



Operational control tree: authorise narrowly; verify recovery. Source: security-backup…fig1.

INTEGRATED ANSWER ORDER

1. State the business rule and conceptual relationship.
2. Translate it into keys, relations and database constraints.
3. Predict the SQL result at the correct row/group grain.
4. Trace concurrency and failure consequences.
5. Place responsibility across layers; choose the data model.
6. Apply least privilege and a tested restore path.

INFO90002 Database Systems and Data Modelling

SELF-TEST DRILLS · ANSWER PATHS · JUDGEMENT CARDS · EXAM-MORNING RETRIEVAL

Full guide + ask Sia about INFO090002

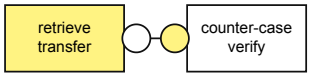
CLOSED-BOOK SELF-TEST

13 · How to drill

PRACTICE

THE FOUR-MOVE LOOP

`Retrieve → transfer → counter-case → verify.` Say the chain without notes; apply it to a changed case; find the first unsupported move; repair that move before starting another case.



Self-test loop: retrieval earns value only when it survives transfer and checking. Source: practice…fig1.

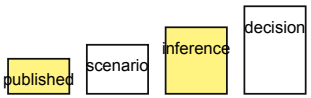
ANSWER SKELETON

1. **Object:** what is being decided, at what lifecycle level?
2. **Rule:** define the operator, dependency, constraint or control.
3. **Mechanism:** trace how the rule changes the case.
4. **Counter-case:** name the nearest alternative or failure path.
5. **Boundary:** state when the conclusion must narrow or change.

IF YOU SEE…	DO THIS FIRST
Narrative noun	Entity test: identity + retained facts
Repeated value	Ask for an FD; do not normalise by appearance
Aggregate	Write row grain and group grain
Concurrent schedule	Trace interleaving to anomaly

14 · Evidence boundary

C1 CONTROL



Do not present a plausible inference as a published fact. Source: assessment-evidence…fig2.

Control: separate what the material establishes, what the scenario supplies, what you infer, and what you recommend. Each arrow must be a checkable mechanism.

MARK-PROTECTION *Answer the operation actually required; familiar terminology is not, by itself, a precondition for using a method.*

14A · Trap board: model

C2–C5

TRAP	CORRECTIVE QUESTION
Noun ⇒ entity	Must instances be identified and facts retained?
Diagram ⇒ requirement	Which business sentence sets the min/max?

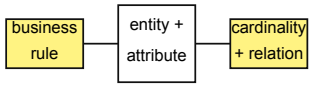
15 · Worked 1: ER judgement

C2 · C3 · 4 MARKS

Prompt: translate stakeholder language into testable requirements and justify every relationship constraint. A draft jumps from business rules straight to relationships.

ANSWER PATH · ONE MARK EACH

① Mark the starting business rule/object. ② Supply the missing entity/attribute mechanism: identifiable instances + retained facts. ③ Reach relationships through stated min/max participation and an identifier. ④ Bound the result: a noun is not automatically an entity; notation cannot resolve ambiguity.



Repair the skipped middle: rule → entity/attribute → justified relation. Source: requirements…fig1.

FULL-CREDIT CONCLUSION

"The rule requires identifiable occurrences and stored facts, so the noun is modelled as an entity with a named identifier. Its relationship uses the stated minimum and maximum. If the stakeholder rule is ambiguous, record the question rather than using diagram notation to invent an answer."

TRAP	REPAIR
Every noun becomes an entity	Apply identity + retained-facts test
Draw first, justify later	Write business sentence first
Associative entity "solves" ambiguity	Return to missing requirement

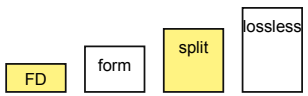
16 · Worked 2: normalisation

C4 · 4 MARKS

Prompt: remove update anomalies while preserving meaning and joinability. A response selects a decomposition merely because "functional dependency" appears.

ANSWER PATH · ONE MARK EACH

① Extract the required outcome. ② State when the claimed **X→Y** holds across all valid tuples. ③ Test the target normal form and the nearest alternative, then check lossless decomposition. ④ Choose/narrow: repeated-looking values alone do not prove a violation.



Normalisation proof path, not a repeated-value reflex. Source: logical-design…fig1.

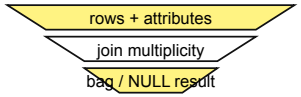
17 · Worked 3: query result

C5 · 4 MARKS

Prompt: predict a query result from relational operations before running SQL. A draft recognises selection/projection but ignores joins and bag behaviour.

ANSWER PATH · ONE MARK EACH

① Extract the required output and starting relation(s). ② Use selection for rows and projection for attributes. ③ Apply the join and inspect matching multiplicity. ④ Check duplicates and NULL/UNKNOWN; narrow any pure-set conclusion that SQL does not preserve.



Work top-down; each narrowing step must preserve the intended grain. Source: relational-algebra…fig1.

QUESTION TO ASK	FAILURE CAUGHT
What is one input row?	Wrong starting grain
How many matches per row?	Join multiplication
Are duplicates removed?	Set assumption imposed on bag result
Can a predicate meet NULL?	TRUE/FALSE assumption misses UNKNOWN

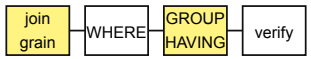
18 · Worked 4: aggregate

C7 · 4 MARKS

Prompt: choose a query shape that preserves row and group grain. A draft chooses grouping/HAVING because the words are familiar.

ANSWER PATH · ONE MARK EACH

① State the exact output/grain. ② Use WHERE only for row predicates; form groups; use HAVING only for group predicates. ③ Test whether a nested query or view better preserves the required abstraction. ④ Inspect join cardinality before trusting the scalar/aggregate.



Aggregate control: join grain precedes filters and grouping. Source: sql-aggregation…fig2.

CLASSIC TRAP *A plausible total may still be inflated. The answer is incomplete until join cardinality is checked.*

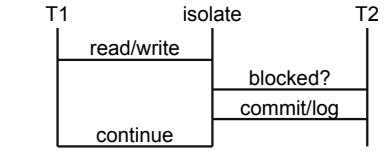
19 · Worked 5: concurrency

C8 · 4 MARKS

Prompt: trace an interleaving to an anomaly and isolation control. A draft jumps from ACID directly to logging/recovery.

ANSWER PATH · ONE MARK EACH

① Mark the starting transaction/invariant. ② Restore the missing locking/isolation mechanism. ③ Show how the control changes the interleaving, then connect logging to recovery. ④ Qualify: serialisable behaviour may impose blocking, retries or deadlocks.

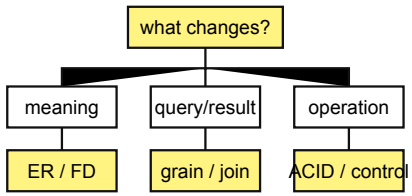


Trace the schedule through the control; name its operational cost. Source: transactions…fig1.

IF YOU SEE…	DO…
Two transactions read/write shared data	Order every operation
Inconsistent/lost result	Name the anomaly and invariant
"Use serialisable"	Explain enforcement + blocking/retry/deadlock cost
Failure after writes	Explain logging/recovery role

20 · Method chooser

C2–C12



New method chooser: classify the required outcome before selecting vocabulary.

CHOICE DISCIPLINE

Meaning problem → business rule/ER/FD. Result problem → relational operations/grain/join. Operational problem → transaction/layer/distribution/security/recovery. Then test the nearest alternative.

20A · Trap board: execution

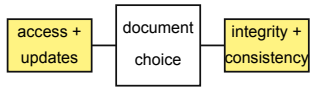
C5–C8

TRAP	CORRECTIVE QUESTION
SQL is pure set logic	What do duplicates and NULL/UNKNOWN do?
Aggregate looks plausible	Did the join multiply rows?
App validates it	Can another writer bypass the app?
Serialisable is always best	What blocking/retry/deadlock cost follows?

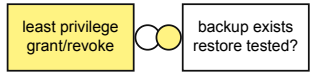
21 · NoSQL + security cards

C11 · C12

IF YOU SEE X…	→ DO Y
"NoSQL is flexible"	Name access pattern, field meaning/types, enforcement boundary and evolution need.
Embed vs reference	Compare access and update boundaries, integrity and consistency.
"Schema-on-read"	Say where enforcement moved; schema still exists.
User/role request	Grant minimum privilege needed; use GRANT/REVOKE.
"There is a backup"	Demand a restore test; a file does not prove recoverability.



Document-model bridge: evidence in; bounded choice out. Source: nosql-json…fig2.



Operational verification: authorisation and recovery answer different risks. Source: security-backup…fig2.

CLASSIC TRAPS

- **Keyword selection:** choosing a document model, architecture or control because its label appears.
- **Flexibility = no rules:** schema-on-read moves enforcement; it does not remove meaning or types.
- **Backup = recoverable:** only a restore test proves the recovery path.
- **Broad privilege by default:** grant only what the stated role needs at the data boundary.

22 · Fast-recall table

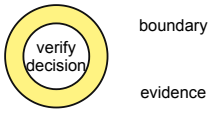
GLOSSARY

TERM	ONE-LINE RECALL
Logical independence	Protect users from schema changes
Physical independence	Protect users from storage changes
Referential integrity	FK matches referenced key or allowed NULL
Transaction	Logical unit of work; should satisfy ACID
Normalisation	Dependency-led decomposition against anomalies
Schema-on-read	Schema remains; only its enforcement boundary moves
Integrated decision	Combine modelling, SQL and operational controls in one defensible database decision; test the failure path

23 · Whole-course trap board

C2–C12

TRAP	CORRECTIVE QUESTION
Distribution solves failure	Which new node/link failure appears?
Warehouse first, grain later	What exactly does one fact row represent?
NoSQL has no schema	Where are meaning/types enforced?
Backup file = safe	Has restoration been tested?



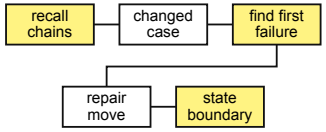
Verification ring: a decision survives mechanism, boundary, counter-case and evidence. Source: practice…fig2.

COMPRESSED JUDGEMENT SENTENCE

"Because the task requires **[outcome]** and the precondition **[condition]** holds, use **[method]**; it changes **[object]** by **[mechanism]**. Reject **[nearest alternative]** because **[discriminator]**. Narrow the claim if **[boundary]**."

24 · Exam-morning plan

STUDY STRATEGY



New retrieval sequence for the final review: recall, transfer, diagnose, repair, qualify.

1. From memory, recite: lifecycle; entity test; cardinality; X→Y; selection/projection/join; WHERE→GROUP→HAVING; ACID/control/recovery.
2. Recite the advanced chain: layers/distribution; fact grain/ETL/lineage; document choice; least privilege; restore test.
3. Take one changed scenario. Classify the required outcome before choosing the method.
4. Speak the answer skeleton: object → rule → mechanism → counter-case → boundary.
5. Record the first unsupported move and repair only that move; then repeat without looking.

FINAL MENTAL CHECKSUM

`Meaning correct? Grain correct? Constraint at the strongest boundary? Failure path traced? Trade-off stated?`

MORNING RULE *Use this page to retrieve, not to reread passively. On exam day the sheet stays out; what matters is whether each chain can be reconstructed and applied.*