Solutions

2.1

- Department – one value per department, so it lives on the one side ✓.

2.2

- `student_id` referencing Student and `course_id` referencing Course ✓.

2.3

- No referencing row may name a target row that does not exist ✓.

3.1

- Entities: Customer, Film, Screening, Ticket ✓.
- One-to-many: Screening to Ticket – `ticket.screening_id` foreign key ✓.
- Many-to-many: Customer to Film through Ticket (a customer watches many films; a film gathers many customers) ✓.
- Key for Ticket: (screening_id, seat_no) identifies one seat at one showing – natural and duplicate-proof ✓.
- Surrogate alternative: a ticket_id is stable if seats get renumbered and carries no meaning to maintain ✓.
- Composite rule: `PRIMARY KEY (screening_id, seat_no)` rejects a second sale of the same seat ✓.

3.2

- It rejects a second row with the same (student, course, term) – the same student enrolling twice in one term ✓.
- Dropped: two racing applications both check-then-insert and both succeed – the engine no longer guarantees it ✓.
- Foreign keys: `student_id` (every enrollment names a real student) and `course_id` (a real course) ✓.