This creates the Campus Housing Services (CHS) schema with tables:

* buildings, rooms, students, applications, contracts, payments, staff, maintenance\_requests, maintenance\_work\_items
* Primary/foreign keys, basic CHECK constraints, and helpful indexes
* Seed data so queries return meaningful results

Schema description in details: <https://docs.google.com/document/d/1DkLwJaZHjGpxdRrMPvIsRcDxiQAjf6HU/edit?usp=sharing&ouid=109057132541194905649&rtpof=true&sd=true>

**Write the SQL statements for the folowing requirements**

**Level 1 —SELECTs (single table)**

1. List all buildings (code, name, year built) ordered by year\_built descending.
2. Show all rooms and their monthly rents in Hawthorn Hall (building\_code='HAW').
3. List student full names and emails for scholarship students only (scholarship\_flag='Y').
4. Show all payments with amount >= 650.
5. Find all applications with status='Pending'.

**Level 2 — Filtering, sorting, simple expressions**

1. List rooms costing under 650 OR with capacity > 1, ordered by rent then room\_code.
2. Show students born after 2005-01-01, sorted by dob.
3. Display payments in October–November 2025 (use payment\_date between).
4. Show applications where intended\_end > intended\_start (sanity check) and compute intended months as (julianday(intended\_end) - julianday(intended\_start))/30.0 (or date diff for your DB).
5. Show distinct room\_type values in the system.

**Level 3 — Basic joins**

1. List each room with its building name and address.
2. For each contract, show student full name, room\_code, contract status, and start\_date/end\_date (join: contracts → applications → students).
3. Show payments with contract status (join: payments → contracts).
4. List maintenance work items with their request’s issue\_category and assigned staff name (multi-join).
5. Show applications with their first preference building name (join on pref1\_building).

**Level 4 — Aggregation & GROUP BY**

1. Count rooms per building; show building name and room count.
2. Average base rent by room type; include min and max.
3. For each contract, total amount paid so far and outstanding balance = months\_between(start\_date,end\_date)\*base\_monthly\_rent - sum(paid) *(hint: you can join contracts→rooms for rent and aggregate payments; for months\_between use DB-specific date math)*.
4. Count applications by status.
5. For each staff role, count assigned maintenance work items.

**Level 5 — HAVING, subqueries, and conditional logic**

1. Show buildings that have **more than 2** single rooms (use HAVING).
2. Find students who have **no payments yet** (subquery or LEFT JOIN … IS NULL to payments through contracts).
3. List rooms whose rent is **above the overall average rent** (subquery).
4. List contracts whose **total paid < deposit\_amount** (join + aggregate + HAVING).
5. For each building, show the **most expensive** room (correlated subquery or window function if available).