
Sample Database Structure

Students Table

student_id	name	course_id	age	fee
1	Alice	101	21	5000
2	Bob	102	22	4500
3	Charlie	101	20	5000
4	David	103	23	4000
5	Eve	102	21	4500

Courses Table

course_id	course_name
101	CS
102	IT
103	EE

---

1. SELECT Statements to Retrieve Data

-- Retrieve all columns SELECT * FROM Students;

-- Retrieve specific columns SELECT name, age FROM Students;

-- Retrieve unique course_ids SELECT DISTINCT course_id FROM Students;

---

2. WHERE Clause

-- Students older than 21 SELECT name, age FROM Students WHERE age > 21;

-- Students older than 20 and fee above 4500 SELECT name, age, fee FROM Students WHERE age > 20 AND fee > 4500;

-- Students whose name starts with 'A' SELECT name FROM Students WHERE name LIKE 'A%';

---

3. ORDER BY Clause

-- Sort by age ascending SELECT name, age FROM Students ORDER BY age ASC;

-- Sort by fee descending SELECT name, fee FROM Students ORDER BY fee DESC;

-- Sort by multiple columns SELECT name, course_id, age FROM Students ORDER BY course_id ASC, age DESC;

---

4. Aggregate Functions (SUM, COUNT, AVG)

-- Count total students SELECT COUNT(*) AS TotalStudents FROM Students;

-- Total fees collected SELECT SUM(fee) AS TotalFees FROM Students;

-- Average student age SELECT AVG(age) AS AverageAge FROM Students;

-- Max and Min fees SELECT MAX(fee) AS MaxFee, MIN(fee) AS MinFee FROM Students;

---

5. GROUP BY and HAVING

-- Count of students per course SELECT course_id, COUNT(*) AS StudentsPerCourse FROM Students GROUP BY course_id;

-- Average fee per course SELECT course_id, AVG(fee) AS AvgFee FROM Students GROUP BY course_id;

-- Courses with more than 1 student SELECT course_id, COUNT*() AS StudentsPerCourse FROM Students GROUP BY course_id HAVING COUNT(*)* > 1;

---

6. Subqueries and Nested Queries

-- Students enrolled in CS course (single-row) SELECT name FROM Students WHERE course_id = (SELECT course_id FROM Courses WHERE course_name = 'CS');

-- Students enrolled in CS or IT (multi-row) SELECT name FROM Students WHERE course_id IN (SELECT course_id FROM Courses WHERE course_name IN ('CS', 'IT'));

-- Students with fee above average of their course (correlated) SELECT s1.name, s1.fee, s1.course_id FROM Students s1 WHERE s1.fee > (SELECT AVG(s2.fee) FROM Students s2 WHERE s2.course_id = s1.course_id);

---

7. Combined Example

-- Total fees per course where total fees > 9000 SELECT course_id, SUM(fee) AS TotalFees FROM Students GROUP BY course_id HAVING SUM(fee) > 9000;

Key Notes

- WHERE filters **rows**; HAVING filters **groups**
- Aggregate functions (`SUM`, `COUNT`, `AVG`) often used with GROUP BY
- Subqueries can be in WHERE, FROM, SELECT
- ORDER BY sorts final result, optionally by multiple columns
- Distinguish **row-level** (WHERE) vs **group-level** (HAVING) filtering