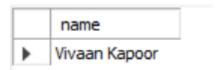
Task 3: SQL for Data Analysis

```
CREATE DATABASE student_management;
USE student_management;
-- Table to store student records
CREATE TABLE students (
  student id INT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  age INT NOT NULL,
  grade FLOAT NOT NULL
);
INSERT INTO students (student_id, name, age, grade) VALUES
(101, 'Aarav Sharma', 18, 85.5),
(102, 'Meera Patel', 17, 91.0),
(103, 'Rohan Mehta', 19, 76.3),
(104, 'Diya Verma', 18, 88.2),
(105, 'Kabir Singh', 17, 67.4),
(106, 'Ananya Joshi', 18, 93.5),
(107, 'Vivaan Kapoor', 19, 81.0),
(108, 'Isha Bhatia', 18, 79.8),
(109, 'Aryan Desai', 17, 84.1),
(110, 'Saanvi Rao', 19, 89.9);
```

1) select name from students where student_id=107;



2) select name, student_id from students order by student_id DESC;

	name	student_id
•	Trisha Kapoor	120
	Ishaan Agarwal	119
	Neha Jain	118
	Parth Gupta	117
	Krisha Shah	116

name	student_id
Aditya Rawat	115
Simran Nair	114
Rajat Chauhan	113
Tanya Malik	112
Yash Khanna	111
name	student_id
Saanvi Rao	110
Aryan Desai	109
Isha Bhatia	108
Vivaan Kapoor	107
Ananya loshi	106
name	student_id
Isha Bhatia	108
Vivaan Kapoor	107
Ananya Joshi	106
Kabir Singh	105
Diva Verma	104

3) SELECT age, MIN(name) AS sample_student FROM students GROUP BY age;

	age	sample_student
•	18	Aarav Sharma
	17	Aryan Desai
	19	Parth Gupta

```
CREATE TABLE courses (
   course_id INT PRIMARY KEY,
   course_name VARCHAR(100) NOT NULL
);
-- Insert some courses
INSERT INTO courses (course_id, course_name) VALUES
(201, 'Mathematics'),
(202, 'Physics'),
(203, 'Computer Science');
```

-- Junction table for student-course enrollment

```
CREATE TABLE enrollments (
  student_id INT,
  course_id INT,
  FOREIGN KEY (student_id) REFERENCES students(student_id),
  FOREIGN KEY (course_id) REFERENCES courses(course_id)
);
-- Insert some enrollments
INSERT INTO enrollments (student_id, course_id) VALUES
(101, 201),
(101, 202),
(102, 201),
(103, 203),
(104, 202),
(106, 203),
(107, 201),
(109, 202),
(110, 203);
(inner join)
SELECT s.name, c.course_name
FROM students s
INNER JOIN enrollments e ON s.student_id = e.student_id
INNER JOIN courses c ON e.course_id = c.course_id;
```

	name	course_name
•	Aarav Sharma	Mathematics
	Meera Patel	Mathematics
	Vivaan Kapoor	Mathematics
	Aarav Sharma	Physics
	Diya Verma	Physics
	name	course_name
	Diya Verma	Physics
	Aryan Desai	Physics
	Rohan Mehta	Computer Science
	Ananya Joshi	Computer Science
	Saanvi Rao	Computer Science

(left join)
SELECT s.name, c.course_name
FROM students s
LEFT JOIN enrollments e ON s.student_id = e.student_id
LEFT JOIN courses c ON e.course_id = c.course_id;

	name	course_name
•	Aarav Sharma	Mathematics
	Aarav Sharma	Physics
	Meera Patel	Mathematics
	Rohan Mehta	Computer Science
	Diva Verma	Physics
	name	course_name
	Kabir Singh	NULL
	Ananya Joshi	Computer Science
	Vivaan Kapoor	Mathematics
	Isha Bhatia	NULL
	Aryan Desai	Physics
	name	course_name
	Saanvi Rao	Computer Science
	Yash Khanna	NULL
	Tanya Malik	NULL
	Rajat Chauhan	NULL
	Simran Nair	NULL
	name	course_name
	Aditya Rawat	NULL
	Krisha Shah	NULL
	Parth Gupta	NULL
	Neha Jain	NULL
	Ishaan Agarwal	NULL
	3	
	Trisha Kapoor	NULL

(right join)
SELECT s.name, c.course_name
FROM courses c
RIGHT JOIN enrollments e ON c.course_id = e.course_id
RIGHT JOIN students s ON e.student_id = s.student_id;

	name	course_name
١	Aarav Sharma	Mathematics
	Aarav Sharma	Physics
	Meera Patel	Mathematics
	Rohan Mehta	Computer Science
	Diva Verma	Physics
	name	course_name
	Kabir Singh	HULL
	Ananya Joshi	Computer Science
	Vivaan Kapoor	Mathematics
	Isha Bhatia	NULL
	Aryan Desai	Physics
	name	course_name
	Saanvi Rao	Computer Science
	Yash Khanna	NULL
	Tanya Malik	NULL
	Rajat Chauhan	NULL
	Simran Nair	HULL
	name	course_name
	Krisha Shah	NULL
	Parth Gupta	NULL
	Neha Jain	NULL
		NULL
	Ishaan Agarwal	

```
(sub queries)
SELECT name, grade
FROM students
WHERE grade > (
SELECT AVG(grade) FROM students
);
```

	name	grade
•	Aarav Sharma	85.5
	Meera Patel	91
	Diya Verma	88.2
	Ananya Joshi	93.5
	Aryan Desai	84.1
	name	grade
	Saanvi Rao	89.9
	Tanya Malik	94.5
	Simran Nair	87.3
	Aditya Rawat	90.2
	Krisha Shah	86
	Neha Jain	92.4
	Trisha Kapoor	88.7

Aggregate function

avg

SELECT AVG(grade) AS average_grade FROM students;

	average_grade
•	83.50500030517578

Sum

SELECT SUM(grade) AS total_grades FROM students;

	total_grades
•	1670.1000061035156

View created

```
CREATE VIEW student_summary_view AS
SELECT
s.student_id,
s.name,
s.age,
s.grade,
(SELECT AVG(s2.grade)
FROM students s2
WHERE s2.age = s.age) AS avg_grade_by_age
FROM students s;

CREATE INDEX idx_students_age ON students(age);

SELECT AVG(grade) FROM students WHERE age = 18;

AVG(grade)
```

86.3999998304579