```sql
-- Retrieves all columns and rows from the "employee" table in the "first_schema".
SELECT * FROM first_schema.employee;

-- Selects employee ID and name from the "employee" table in the "first_schema".
SELECT emp_id, emp_name FROM first_schema.employee;

-- Fetches all records from "employee" where the employee's location is Maharashtra.
SELECT * FROM first_schema.employee WHERE location = "maharashtra";

-- Selects employee names from the "employee" table where the location is Maharashtra and
the job domain is marketing.
SELECT emp_name FROM first_schema.employee WHERE location = "maharashtra" AND jo
b_domain = "marketing";

-- Lists distinct locations of employees in the "employee" table.
SELECT DISTINCT location FROM first_schema.employee;

-- Selects all records where the employee's name contains the letter 's'.
SELECT * FROM first_schema.employee WHERE emp_name LIKE '%s%';

-- Retrieves all records from "employee" where the score is greater than 8 and the job domain
 is marketing.
SELECT * FROM first_schema.employee WHERE score > 8 AND job_domain='marketing';

-- Selects all records from "employee" where the score is between 6 and 9, sorted in descend
ing order by score.
SELECT * FROM first_schema.employee WHERE score >=6 AND score <=9 ORDER BY sc
ore DESC;

-- Fetches all records from "employee" where the score is between 6 and 9 and job domain is
 sales, ordered by score.
SELECT * FROM first_schema.employee WHERE score BETWEEN 6 AND 9 AND job_domain
 = 'sales' ORDER BY score;

-- Selects the first 5 records from "employee" where location is Maharashtra.
SELECT * FROM first_schema.employee WHERE location = "maharashtra" LIMIT 5;

-- Retrieves 5 records after skipping the first record from "employee" where the location is
Maharashtra and score is over 8, ordered by score descending.
SELECT * FROM first_schema.employee WHERE location = "maharashtra" AND score >8
ORDER BY score DESC LIMIT 5 OFFSET 1;

-- Counts the number of employees in Maharashtra.
SELECT COUNT(*) FROM first_schema.employee WHERE location = "maharashtra";

-- Retrieves the maximum score of employees from Maharashtra in the sales domain.
SELECT MAX(score) FROM first_schema.employee WHERE location = "maharashtra" AND jo
b_domain = "sales";

-- Retrieves the minimum score of employees from Maharashtra in the sales domain.
SELECT MIN(score) FROM first_schema.employee WHERE location = "maharashtra" AND jo
b_domain = "sales";

-- Calculates the average score of employees from Maharashtra in the sales domain.
SELECT AVG(score) FROM first_schema.employee WHERE location = "maharashtra" AND jo
b_domain = "sales";

-- Calculates the rounded average score of employees from Maharashtra in the sales domain
to one decimal place.
SELECT ROUND(AVG(score),1) FROM first_schema.employee WHERE location = "maharasht
ra" AND job_domain = "sales";
```

```sql
-- Retrieves the maximum, minimum, and rounded average score of employees in the marketing domain in Punjab.
SELECT MAX(score) AS max_score,
MIN(score) AS min_score,
ROUND(AVG(score),1) AS avg
FROM first_schema.employee
WHERE job_domain = "marketing" AND location = "punjab";

-- Counts the number of employees in each location.
SELECT location, COUNT(location) FROM first_schema.employee GROUP BY location;

-- Retrieves the job domain, count of job domains, and rounded average score for each job domain, sorted by average score in descending order.
SELECT job_domain, COUNT(job_domain) AS domain_count, ROUND(AVG(score),1) AS avg_score
FROM first_schema.employee
GROUP BY job_domain
ORDER BY avg_score DESC;

-- Counts the number of employees joining on the same date where the count is greater than 1, ordered by count descending.
SELECT joining_date, COUNT(*) AS date_count FROM first_schema.employee GROUP BY joining_date HAVING date_count>1 ORDER BY date_count DESC;

-- Counts the number of employees in each job domain where the count is greater than 1, ordered by count descending.
SELECT job_domain, COUNT(*) AS job_domain_count FROM first_schema.employee GROUP BY job_domain HAVING job_domain_count>1 ORDER BY job_domain_count DESC;

-- Determines the job status based on the job domain, marking those in "Sales" as "Promoted".
SELECT emp_name, IF(job_domain="Sales","Promoted","Next Time Pakka") AS job_status
FROM first_schema.employee;

-- Selects employee names where the name starts with 's_'.
SELECT emp_name FROM first_schema.employee WHERE emp_name LIKE "s_%";

-- Creates a new database named "student".
CREATE DATABASE student;

-- Creates a new table named "academics" in the "student" database with specified columns and primary key.
CREATE TABLE student.academics(
student_id INT,
naame VARCHAR(250),
remarks VARCHAR(250),
PRIMARY KEY (student_id)
);

-- Creates a new table named "sports" in the "student" database with specified columns, a primary key, and a foreign key reference.
CREATE TABLE student.sports(
sports_id INT,
naame VARCHAR(250),
student_id INT,
PRIMARY KEY (sports_id),
FOREIGN KEY (student_id) REFERENCES academics(student_id)
);

-- Inserts sample data into the "academics" table in the "student" database.
INSERT INTO student.academics
VALUES (1, "A", "GOOD"),
(2, "A", "BAD");
```