Sample Questions on SQL FA 2025

The relational schema for student course registration are as follows

Student (Id, name, street, city, dept_id, Mobile, email, CGPA, Fee-paid, tot_cred)
Takes (course-id, id, semester, year, GP, marks, course_fee)
Course (course-id, title, credit-hour)
Student_backup (Id, name, street, city, dept_id, Mobile, email, CGPA, Fee-paid, tot_cred)

Single table

Q. 1: Write SQL statements for the following queries

a. Find all distinct street and cities where students live.
b. Find all cities for students with Ids between 1111 and 6666. There will be no duplicate in the result.
```
WHERE id BETWEEN 1111 AND 6666;
```
c. Find Id, name, street and city of all students who have no email.
```
WHERE email IS NULL
```
d. Find Id, name, street and city of all students who use Gmail as his email. You have to search sub-string "gmail".
```
WHERE LOWER(email) LIKE '%gmail%';
```

Multiple tables queries

Q2:

```
SELECT s.id, s.name, t.course_id
FROM student s, takes t
WHERE s.id = t.id
  AND t.year = 2000;
```

```
SELECT s.id, s.name, t.course_id, c.title
FROM student s, takes t, course c
WHERE s.id = t.id
  AND t.course_id = c.course_id
  AND c.credit_hour = 3;
```

a. Find all Id, name and course id of all students who took courses in the year 2000.
b. Find all Id, name, course id, title of all students who registered 3 credit hour courses.

employee (*person-name, street, city*)
works (*person-name, company-name, salary*)
company (*company-name, city*)

```
SELECT e.person_name, e.street, e.city AS employee_city,
       w.company_name, c.city AS company_city
FROM employee e, works w, company c
WHERE e.person_name = w.person_name
  AND w.company_name = c.company_name
  AND w.salary > 10000;
```

Write SQL for the following queries:
c. Find person name, street, employee city, company name, company city for all employees salary greater than 10000.

Single table group by

```
SELECT semester, year, AVG(marks) AS avg_marks
FROM takes
WHERE marks > 50
GROUP BY semester, year
HAVING AVG(marks) > 60
ORDER BY year DESC;
```

Q. 3: Write SQL statements for the following queries:

a. Find the city and street wise maximum, minimum and average CGPA for all students with CGPA >= 2 and average CGPA is also greater than or equal to 2.

```
SELECT semester, year, AVG(marks) AS avg_marks
FROM takes
WHERE marks > 50
GROUP BY semester, year
HAVING AVG(marks) > 60
ORDER BY year DESC;
```

b. Find semester, year wise average marks for all students with marks higher than 50 and average marks higher than 60 in descending order of year.

Multiple table group by

Student (Id, name, street, city, dept_id, Mobile, email, CGPA, Fee-paid, tot_cred)
Takes (course-id, id, semester, year, GP, marks, course_fee)
Course (course-id, title, credit-hour)

Q4: Student_backup (Id, name, street, city, dept_id, Mobile, email, CGPA, Fee-paid, tot_

a. Write SQL to find semester, year wise average marks of all students of 'Dhaka' or 'Khulna' city with average marks higher than 50.

```
s.id = t.id
```

b. Find the city and street wise total amount, maximum, minimum and average Fee-paid in year 2016 in descending order of total fee-paid for students with fee-paid higher than 100000 and average fee-paid greater than 200000.

```
s.id = t.id
```

c. Find the city and street wise total number of students, maximum, minimum and average CGPA for all 3 credit-hour courses. Output will be city, street, total-num-student, max-CGPA, min-CGPA, avg-CGPA.

```
s.id = t.id
  AND t.course_id = c.course_id
```

Subquery

Q5:

a. Find id, name, street, city and CGPA of those students who have taken courses in spring 2018 and CGPA is greater than the average GP of students of Dhaka district.

```
SELECT id, name,
street, city, CGPA
FROM student
WHERE id IN (
   SELECT id
   FROM takes
   WHERE semester =
'Spring'

AND year = 2018
)
AND CGPA > (
   SELECT AVG(GP)
   FROM takes
   WHERE id IN (
      SELECT id
      FROM student
      WHERE city =
'Dhaka'
      )
   )
```

Q. 6: For the given queries as follows, find the schema and write SQL:

branch(*branch-name*, *branch city*, *assets*)
customer (*customer-name*, *customer-street*, *customer-city*)
loan (*loan-number*, *branch-name*, *amount*)
borrower (*customer-name*, *loan-number*)
account (*account-number*, *branch-name*, *balance*)
depositor (*customer-name*, *account-number*)

a. Find customer name, customer street and customer city of all customers who has same amount of loan as customer 'Abid'.

```
borrower.loan_number = loan.loan_number
```

b. Find customer name, branch name and amount of loan for all customers who have loan in the same branches where 'Zahid' has loan.

```
l.loan_number = b.loan_number
```

c. Find customer name, account number and customer city of all customers who has account in the same branches where 'Zahid' has accounts.

d. Find the list of customer name, branch name and branch city of all customers who have accounts in all branch city.

```
WHERE depositor.account_number =
account.account_number
  AND depositor.customer_name =
customer.customer_name
```

```
a.account_number = d.account_number
      AND a.branch_name = b.branch_name
      AND d.customer_name = c.customer_name
```

e. Find customer name, street and city of those customers who have accounts at all branches located in 'Barishal'.

```
WHERE a.account_number = d.account_number
      AND a.branch_name = b.branch_name
      AND d.customer_name = c.customer_name
```

employee (*person-name*, *street*, *city*)
works (*person-name*, *company-name*, *salary*)
company (*company-name*, *city*)

```
SELECT person_name
FROM employee
WHERE city IN (
   SELECT city
   FROM company
);
```

Write SQL for the following queries:

f. Find person name, street and city of all employees who live in the same city as 'Mr. Akib' lives.

g. Find all person name of all employees who live in the same city as the company.

Insert, Delete and Update

Q7:

branch(branch-name, branch city, assets)
customer (customer-name, customer-street, customer-city)
loan (loan-number, branch-name, amount)
borrower (customer-name, loan-number)
account (account-number, branch-name, balance)
depositor (customer-name, account-number)

a. Insert all the loans of branch name = "NSU" to the account relation as loan number will be account number and amount will be balance.

b. Update the database as follows: those borrower have loans more than 50000, decrease their loan by 10%. For other borrowers, make their loan zero.

c. Delete all loans of customers who lives in 'Gazipur'.

d. Update fee-paid for all students by the total course-fee paid for all taken courses and course fee is not null. takes.id = student.id

e. Insert all students of Spring 2018 into student_backup relation and delete their information from takes and student relation.

f. Make course fee = 0 for all students taking course in Summer 2024 and CGPA =4.

g. Update total credit of all students with the sum of credit hour of courses taken and GP not null and not 'F'

```
WHERE takes.course_id = course.course_id
    AND takes.id = student.id
```

Student (Id, name, street, city, dept_id, Mobile, email, CGPA, Fee-paid, tot_cred)
Takes (course-id, id, semester, year, GP, marks, course_fee)
Course (course-id, title, credit-hour)
Student_backup (Id, name, street, city, dept_id, Mobile, email, CGPA, Fee-paid, tot_cred)