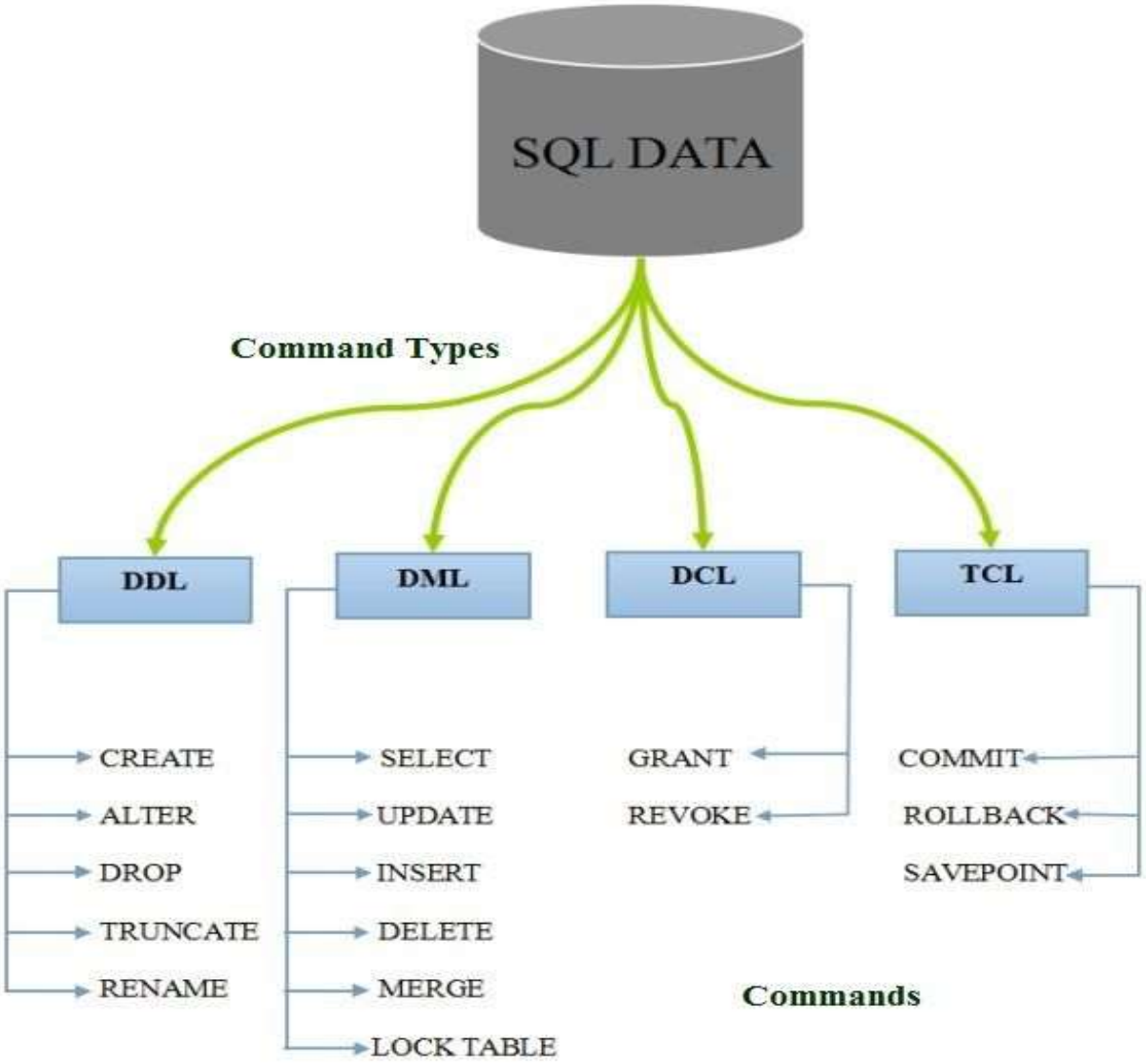
SQL Commands

SQL BASIC

By JAHANGIR HUSSEN



```sql
CREATE TABLE Employee (
    name VARCHAR(50),
    id INT,
    Department VARCHAR(50),
    age INT,
    salary INT,
    city VARCHAR(50)
);
INSERT INTO Employee (name, id, Department, age, salary, city)
VALUES
    ('Jamal', 1, 'IT', 25, 50000, 'Dhaka'),
    ('Jeams', 2, 'Finance', 30, 60000, 'Rajshahi'),
    ('Munni', 3, 'Marketing', NULL, 45000, 'Khulna'),
    ('Sadia', 4, 'HR', 28, 55000, 'Chittagong'),
    ('Sonia', 5, 'Operations', 28, 55000, 'Chittagong'),
    ('Salman', 6, 'Sales', NULL, 52000, 'Sylhet');
```

```sql
SELECT name, id, Department, age, salary, city
FROM Employee;
```

name	id	Department	age	salary	city
Jamal	1	IT	25	50000	Dhaka
Jeams	2	Finance	30	60000	Rajshahi
Munni	3	Marketing	NULL	45000	Khulna
Sadia	4	HR	28	55000	Chittagong
Sonia	5	Operations	28	55000	Chittagong
Salman	6	Sales	NULL	52000	Sylhet

1)আপনি যদি নতুন ডেটা টুকুই টেবিলে যোগ করতে চান, **INSERT INTO**

```sql
INSERT INTO Employee (name, id, Department, age, salary, city)
VALUES
    ('Rahim', 7, 'IT', 26, 52000, 'Dhaka'),
    ('Karim', 8, 'Finance', 35, 70000, 'Chittagong');
```

SELECT কোয়েরি ব্যবহার করলে আউটপুট হবে

name	id	Department	age	salary	city
Jamal	1	IT	25	50000	Dhaka
Jeams	2	Finance	30	60000	Rajshahi
Munni	3	Marketing	NULL	45000	Khulna
Sadia	4	HR	28	55000	Chittagong
Sonia	5	Operations	28	55000	Chittagong
Salman	6	Sales	NULL	52000	Sylhet
Rahim	7	IT	26	52000	Dhaka
Karim	8	Finance	35	70000	Chittagong

2)সকল ডেটা বা কোনও নির্দিষ্ট শর্তের মোতাবেক কিছু ডেটা মুছতে চান, তবে **DELETE**

```sql
DELETE FROM Employee
WHERE Department = 'Marketing';
```

SELECT কোয়েরি ব্যবহার করলে আউটপুট হবে

name	id	Department	age	salary	city
Jamal	1	IT	25	50000	Dhaka
Jeams	2	Finance	30	60000	Rajshahi
Sadia	4	HR	28	55000	Chittagong
Sonia	5	Operations	28	55000	Chittagong
Salman	6	Sales	NULL	52000	Sylhet
Rahim	7	IT	26	52000	Dhaka
Karim	8	Finance	35	70000	Chittagong

3)নির্দিষ্ট সারির ডেটা আপডেট করতে চান, তবে **UPDATE**

```
UPDATE Employee
SET salary = 60000
WHERE id = 1;
```

SELECT কোয়েরি ব্যবহার করলে আউটপুট হবে

name	id	Department	age	salary	city
Jamal	1	IT	25	60000	Dhaka
Jeams	2	Finance	30	60000	Rajshahi
Munni	3	Marketing	NULL	45000	Khulna
Sadia	4	HR	28	55000	Chittagong
Sonia	5	Operations	28	55000	Chittagong
Salman	6	Sales	NULL	52000	Sylhet
Rahim	7	IT	26	52000	Dhaka
Karim	8	Finance	35	70000	Chittagong

4) **WHERE** বৈশিষ্ট্যটি ব্যবহার হয় ডেটাবেস কোয়েরিতে নির্দিষ্ট শর্ত সাধনের জন্য

```
SELECT * FROM Employee
WHERE Department = 'IT';
```

এই কোডটি 'IT' বিভাগের সকল কর্মচারীর ডেটা দেখাবে

name	id	Department	age	salary	city
Jamal	1	IT	25	60000	Dhaka
Rahim	7	IT	26	52000	Dhaka

UPDATE স্টেটমেন্টে WHERE ব্যবহার:

```
UPDATE Employee
SET salary = 65000
WHERE id = 2;
```

DELETE স্টেটমেন্টে WHERE ব্যবহার:

```
DELETE FROM Employee
WHERE age IS NULL;
```

SELECT কোয়েরি ব্যবহার করলে আউটপুট হবে

name	id	Department	age	salary	city
Jamal	1	IT	25	60000	Dhaka
Jeams	2	Finance	30	65000	Rajshahi
Sadia	4	HR	28	55000	Chittagong
Sonia	5	Operations	28	55000	Chittagong
Rahim	7	IT	26	52000	Dhaka
Karim	8	Finance	35	70000	Chittagong

7) এটি ডুপ্লিকেট মানগুলি সরানোর জন্য ব্যবহৃত হয়।

```
SELECT DISTINCT Department
FROM Employee;
```

5) **GROUP BY** ব্যবহার করে ডেটাবেস টেবিলের ডেটা গুলি একই গুলি মান অনুসারে গ্রুপ করতে ব্যবহৃত হয়। এটি সাধারণভাবে গণনা, যোগফল, সর্টিং ইত্যাদি করতে ব্যবহৃত হয়

```
SELECT Department, COUNT(*) AS EmployeeCount, AVG(salary) AS
AvgSalary
FROM Employee
GROUP BY Department;
```

Department	EmployeeCount	AvgSalary
IT	2	56000
Finance	2	67500
HR	1	55000
Operations	1	55000
Sales	1	52000

6) **ORDER BY** ব্যবহার করে ডেটাবেস টেবিলের ডেটা গুলি একই গুলি মান অনুসারে সাজানো হয়। এটি ডেটা সর্ট করতে ব্যবহৃত হয়।

```
SELECT * FROM Employee
ORDER BY salary DESC;
```

name	id	Department	age	salary	city
Karim	8	Finance	35	70000	Chittagong
Jeams	2	Finance	30	65000	Rajshahi
Jamal	1	IT	25	60000	Dhaka
Sadia	4	HR	28	55000	Chittagong
Sonia	5	Operations	28	55000	Chittagong
Rahim	7	IT	26	52000	Dhaka
Salman	6	Sales	NULL	52000	Sylhet
Munni	3	Marketing	NULL	45000	Khulna

একাধিক কলামের উপরে ভিত্তি করে সর্ট:

```
SELECT name, salary
FROM Employee
ORDER BY salary DESC, name ASC;
```

কর্মচারীদের বেতন অনুসারে ডেসেন্ডিং এবং তারপরে নাম অনুসারে আসবে

name	salary
Karim	70000
Jeams	65000
Jamal	60000
Sonia	55000
Sadia	55000
Rahim	52000
Salman	52000
Munni	45000

LIMIT ব্যবহার করে নির্দিষ্ট সংখ্যক সারি প্রদর্শন:

```
SELECT name, salary
FROM Employee
ORDER BY salary DESC
LIMIT 3;
```

এই কোডটি দ্বারা ডিপার্টমেন্ট কলামের ডুপ্লিকেট মানগুলি সরিয়ে দেখানো হয়েছে

```
| Department |
|------------|
| IT         |
| Finance    |
| Marketing  |
| HR         |
| Operations |
| Sales      |
```

একাধিক কলামের জন্য DISTINCT:

```
SELECT DISTINCT Department, city
FROM Employee;
```

এই কোডটি দ্বারা ডিপার্টমেন্ট এবং শহর কলামের সময়ক্রমে ডুপ্লিকেট মানগুলি সরিয়ে দেখানো হয়েছে:

```
| Department | city       |
|------------|------------|
| IT         | Dhaka      |
| Finance    | Rajshahi   |
| Marketing  | Khulna     |
| HR         | Chittagong |
| Operations | Chittagong |
| Sales      | Sylhet     |
```

কিছু ক্যারেক্টারের মাধ্যমে DISTINCT:

```
SELECT DISTINCT LEFT(name, 1) AS FirstLetter
FROM Employee;
```

এই কোডটি দ্বারা নাম কলামের প্রথম অক্ষরগুলির সময়ক্রমে ডুপ্লিকেট মানগুলি সরিয়ে দেখানো হয়েছে

```
| FirstLetter |
|-------------|
| J           |
| K           |
| M           |
| S           |
| R           |
```

8) **AND ব্যবহার:**

```
SELECT *
FROM Employee
WHERE Department = 'IT' AND age < 30;
```

এই কোডটি দ্বারা 'IT' বিভাগের এবং 30 বছরের কম বয়সের কর্মচারীদের তালিকা দেখানো হয়েছে

```
| name  | id | Department | age | salary | city  |
|-------|----|------------|-----|--------|-------|
| Jamal | 1  | IT         | 25  | 60000  | Dhaka |
| Rahim | 7  | IT         | 26  | 52000  | Dhaka |
```

এই কোডটি দ্বারা বেতন অনুসারে ডেসেন্ডিং অর্ডারে সর্ট করা হয়েছে এবং তারপরে শুধুমাত্র প্রথম 3 টি সারি দেখানো হয়েছে

```
| name   | salary |
|--------|--------|
| Karim  | 70000  |
| Jeams  | 65000  |
| Jamal  | 60000  |
```

NULL মানগুলি শেজানো:

```
SELECT name, age
FROM Employee
ORDER BY age NULLS LAST;
```

এই কোডটি দ্বারা বয়স অনুসারে সর্ট করা হয়েছে, কিন্তু NULL মানগুলি শেজানো হয়েছে

```
| name   | age |
|--------|-----|
| Jamal  | 25  |
| Rahim  | 26  |
| Sadia  | 28  |
| Sonia  | 28  |
| Jeams  | 30  |
| Karim  | 35  |
| Munni  |     |
| Salman |     |
```

9) **OR ব্যবহার:**

```
SELECT *
FROM Employee
WHERE Department = 'Sales' OR Department = 'Marketing';
```

এই কোডটি দ্বারা 'Sales' অথবা 'Marketing' বিভাগের কর্মচারীদের তালিকা দেখানো হয়েছে

```
| name   | id | Department | age | salary | city    |
|--------|----|------------|-----|--------|---------|
| Munni  | 3  | Marketing  | NULL| 45000  | Khulna  |
| Salman | 6  | Sales      | NULL| 52000  | Sylhet  |
```

11) **HAVING ব্যবহার:**

```
SELECT Department, AVG(salary) AS AvgSalary
FROM Employee
GROUP BY Department
HAVING AVG(salary) > 55000;
```

এই কোডটি দ্বারা ডিপার্টমেন্ট ভিত্তিক কর্মচারীদের গড় বেতন হতে 55,000 টাকা বেশি তাদের তালিকা দেখানো হয়েছে

```
| Department | AvgSalary |
|------------|-----------|
| Finance    | 67500     |
| IT         | 56000     |
```

12) **AS ব্যবহার:**

```
SELECT name, salary * 0.1 AS Bonus
FROM Employee;
```

এই কোডটি দ্বারা কর্মচারীদের বেতনের 10% কে বোনাস হিসেবে দেখানো হয়েছে

```
| name   | Bonus |
|--------|-------|
| Jamal  | 6000  |
| Jeams  | 6500  |
| Munni  | 4500  |
| Sadia  | 5500  |
| Sonia  | 5500  |
| Salman | 5200  |
| Rahim  | 5200  |
| Karim  | 7000  |
```

13) **AVG, MAX, MIN ব্যবহার:**

```
SELECT Department, AVG(salary) AS AvgSalary, MAX(salary)
AS MaxSalary, MIN(salary) AS MinSalary
FROM Employee
GROUP BY Department;
```

এই কোডটি দ্বারা ডিপার্টমেন্ট ভিত্তিক কর্মচারীদের গড়, সর্বাধিক, এবং সর্বনিম্ন বেতন দেখানো হয়েছে

```
| Department | AvgSalary | MaxSalary | MinSalary |
|------------|-----------|-----------|-----------|
| Finance    | 67500     | 70000     | 65000     |
| IT         | 56000     | 60000     | 52000     |
| Marketing  | 45000     | 45000     | 45000     |
| HR         | 55000     | 55000     | 55000     |
| Operations | 55000     | 55000     | 55000     |
| Sales      | 52000     | 52000     | 52000     |
```

10) **NOT ব্যবহার:**

```
SELECT *
FROM Employee
WHERE NOT Department = 'IT';
```

এই কোডটি দ্বারা 'IT' বিভাগের কর্মচারীদের ছাড়া বাকি সব কর্মচারীদের তালিকা দেখানো হয়েছে

```
| name   | id | Department | age | salary | city       |
|--------|----|------------|-----|--------|------------|
| Jeams  | 2  | Finance    | 30  | 65000  | Rajshahi   |
| Munni  | 3  | Marketing  | NULL| 45000  | Khulna     |
| Sadia  | 4  | HR         | 28  | 55000  | Chittagong |
| Sonia  | 5  | Operations | 28  | 55000  | Chittagong |
| Salman | 6  | Sales      | NULL| 52000  | Sylhet     |
| Rahim  | 7  | IT         | 26  | 52000  | Dhaka      |
| Karim  | 8  | Finance    | 35  | 70000  | Chittagong |
```

14) **SUM ব্যবহার:**

```
SELECT Department, SUM(salary) AS TotalSalary
FROM Employee GROUP BY Department;
```

এই কোডটি দ্বারা ডিপার্টমেন্ট ভিত্তিক কর্মচারীদের মোট বেতন দেখানো হয়েছে

```
| Department | TotalSalary |
|------------|-------------|
| Finance    | 137500      |
| IT         | 112000      |
| Marketing  | 45000       |
| HR         | 55000       |
| Operations | 55000       |
| Sales      | 52000       |
```

15) **COUNT ব্যবহার:**

```
SELECT Department, COUNT(*) AS EmployeeCount
FROM Employee
GROUP BY Department;
```

এই কোডটি দ্বারা ডিপার্টমেন্ট ভিত্তিক কর্মচারীদের সংখ্যা দেখানো হয়েছে

```
| Department | EmployeeCount |
|------------|---------------|
| Finance    | 2             |
| IT         | 2             |
| Marketing  | 1             |
| HR         | 1             |
| Operations | 1             |
| Sales      | 1             |
```

16) **IN ব্যবহার:**

```
SELECT *
FROM Employee
WHERE Department IN ('IT', 'Finance');
```

এই কোডটি দ্বারা 'IT' অথবা 'Finance' বিভাগের কর্মচারীদের তালিকা দেখানো হয়েছে:

18) **BETWEEN ব্যবহার:**

```
SELECT * FROM Employee
WHERE age BETWEEN 25 AND 30;
```

25 থেকে 30 বছর কর্মচারীদের তালিকা দেখানো হয়েছে:

name	id	Department	age	salary	city
Jamal	1	IT	25	50000	Dhaka
Rahim	7	IT	26	52000	Dhaka
Sadia	4	HR	28	55000	Chittagong
Sonia	5	Operations	28	55000	Chittagong

19) **LIKE ব্যবহার:**

```
SELECT *
FROM Employee
WHERE name LIKE 'J%';
```

'J' দিয়ে শুরু হওয়া নামের কর্মচারীদের তালিকা:

name	id	Department	age	salary	city
Jamal	1	IT	25	50000	Dhaka
Jeams	2	Finance	30	60000	Rajshahi

20) **UNION ব্যবহার:**

```
SELECT name, id FROM Employee
WHERE Department = 'IT'
UNION
SELECT name, id FROM Employee
WHERE Department = 'Finance';
```

এই কোডটি দ্বারা 'IT' এবং 'Finance' বিভাগের কর্মচারীদের নাম এবং আইডি একত্রে দেখানো হয়েছে:

name	id
Jamal	1
Rahim	7
Jeams	2
Karim	8

21) **OFFSET ব্যবহার:**

```
SELECT name, age FROM Employee
ORDER BY age
LIMIT 3 OFFSET 2;
```

এখানে, **OFFSET** ব্যবহার করে বয়স অনুযায়ী তালিকার শুরু হয় 2 তম কর্মচারী হতে থাকা নাম এবং বয়স দেখানো হচ্ছে:

name	age
Sadia	28
Sonia	28
Rahim	26

name	id	Department	age	salary	city
Jamal	1	IT	25	50000	Dhaka
Jeams	2	Finance	30	60000	Rajshahi
Rahim	7	IT	26	52000	Dhaka
Karim	8	Finance	35	70000	Chittagong

17) **IS NULL ব্যবহার:**

```
SELECT *
FROM Employee WHERE age IS NULL;
```

বয়স তথ্য উপস্থিতনা থাকা কর্মচারীদের তালিকা দেখানো হয়েছে:

name	id	Department	age	salary	city
Munni	3	Marketing		45000	Khulna
Salman	6	Sales		52000	Sylhet

22) **CASE, WHEN, THEN, ELSE ব্যবহার:**

```
SELECT name, salary,
       CASE
         WHEN salary > 60000 THEN 'High Salary'
         WHEN salary > 50000 THEN 'Medium Salary'
         ELSE 'Low Salary'
       END AS SalaryCategory
FROM Employee;
```

এই কোডটি দ্বারা কর্মচারীদের বেতনের উপর ভিত্তি করে তাদের বেতনের শ্রেণি তৈরি করা হয়েছে:

name	salary	SalaryCategory
Jamal	50000	Low Salary
Jeams	60000	Medium Salary
Munni	45000	Low Salary
Sadia	55000	Medium Salary
Sonia	55000	Medium Salary
Salman	52000	Medium Salary
Rahim	52000	Medium Salary
Karim	70000	High Salary

23) **EXISTS ব্যবহার:**

```
SELECT *
FROM Employee e
WHERE EXISTS (
   SELECT 1
   FROM Department d
   WHERE d.id = e.id
);
```

এই কোডটি দ্বারা ডিপার্টমেন্ট তালিকা ও কর্মচারীদের তালিকা একে অপরের সাথে সম্পর্কিত আছে কিনা তা দেখানো হয়েছে:

name	id	Department	age	salary	city
Jamal	1	IT	25	50000	Dhaka

23) **ANY ব্যবহার:**

```
SELECT name, salary FROM Employee
WHERE salary > ANY (SELECT salary FROM Employee WHERE
Department = 'Finance');
```

বিভাগের কর্মচারীদের মধ্যে যে কর্মচারীর বেতনের চেয়ে বৃদ্ধি বেতন আছে তাদের নাম এবং বেতন দেখানো হচ্ছে:

```
| name   | salary |
|--------|--------|
| Jeams  | 60000  |
| Sadia  | 55000  |
| Sonia  | 55000  |
| Salman | 52000  |
| Rahim  | 52000  |
| Karim  | 70000  |
```

24) **INNER JOIN ব্যবহার:**

```
SELECT Employee.name, Department.department_name FROM
Employee INNER JOIN Department ON Employee.department_id
= Department.id;
```

INNER JOIN ব্যবহার করে কর্মচারীর তালিকা এবং তাদের বিভাগের তালিকা মিলানো হয়েছে:

```
| name   | department_name  |
|--------|------------------|
| Jamal  | IT               |
| Jeams  | Finance          |
| Munni  | Marketing        |
| Sadia  | HR               |
| Sonia  | Operations       |
| Salman | Sales            |
| Rahim  | IT               |
| Karim  | Finance          |
```

25) **LEFT JOIN ব্যবহার:**

```
SELECT Employee.name, Department.department_name FROM
Employee LEFT JOIN Department ON Employee.department_id =
Department.id;
```

তালিকা মিলানো হয়েছে, তবে যদি বিভাগ না থাকে তবে এও দেখানো

```
| name   | department_name  |
|--------|------------------|
| Jamal  | IT               |
| Jeams  | Finance          |
| Munni  | Marketing        |
| Sadia  | HR               |
| Sonia  | Operations       |
| Salman | Sales            |
| Rahim  | IT               |
| Karim  | Finance          |
| Munni  | NULL             |
```

```
| Jeams  | 2  | Finance    | 30  | 60000  | Rajshahi |
| Rahim  | 7  | IT         | 26  | 52000  | Dhaka    |
| Karim  | 8  | Finance    | 35  | 70000  | Chittagong|
```

22) **ALL ব্যবহার:**

```
SELECT name, age FROM Employee
WHERE age > ALL (SELECT age FROM Employee WHERE Department =
'Finance');
```

ALL ব্যবহার করে একই বিভাগের সকল কর্মচারীদের বয়সের চেয়ে বৃদ্ধি বয়স আছে তাদের নাম এবং বয়স দেখানো হচ্ছে:

```
| name   | age |
|--------|-----|
| Jeams  | 30  |
| Munni  |     |
| Karim  | 35  |
```

26) **RIGHT JOIN ব্যবহার:**

```
SELECT Employee.name, Department.department_name
FROM Employee
RIGHT JOIN Department ON Employee.department_id = Department.id;
```

RIGHT JOIN ব্যবহার করে বিভাগের তালিকা এবং তাদের সাথে যেকোনো কর্মচারীর তালিকা মিলানো হয়েছে, তবে কোনো কর্মচারী না থাকলেও বিভাগের তালিকা দেখানো হয়েছে:

```
| name   | department_name  |
|--------|------------------|
| Jamal  | IT               |
| Jeams  | Finance          |
| Munni  | Marketing        |
| Sadia  | HR               |
| Sonia  | Operations       |
| Salman | Sales            |
| Rahim  | IT               |
| Karim  | Finance          |
| NULL   | Logistics        |
| NULL   | Public Relations |
```

27) **FULL OUTER JOIN ব্যবহার**

```
SELECT Employee.name, Department.department_name FROM Employee
FULL OUTER JOIN Department ON Employee.department_id =
Department.id;
```

কর্মচারীর তালিকা এবং তাদের সাথে যেকোনো বিভাগের তালিকা মিলানো হয়েছে, তবে যদি কোনো কর্মচারী না থাকে অথবা কোনো বিভাগ না থাকে তবে সেই ক্ষেত্রে পুরোটাই দেখানো হয়েছে:

```
| name   | department_name  |
|--------|------------------|
| Jamal  | IT               |
| Jeams  | Finance          |
| Munni  | Marketing        |
| Sadia  | HR               |
| Sonia  | Operations       |
| Salman | Sales            |
```

29) **Cross Join ব্যবহার:**

```
SELECT Employee.name, Department.department_name
FROM Employee
CROSS JOIN Department;
```

CROSS JOIN ব্যবহার করে কর্মচারীদের তালিকা এবং তাদের সাথে সকল বিভাগের তালিকা মিলানো হয়েছে:

```
| name    | department_name |
|---------|-----------------|
| Jamal   | IT              |
| Jeams   | Finance         |
| Munni   | Marketing       |
| Sadia   | HR              |
| Sonia   | Operations      |
| Salman  | Sales           |
| Rahim   | IT              |
| Karim   | Finance         |
| Jamal   | Logistics       |
| Jeams   | Public Relations |
| Munni   | Logistics       |
| Sadia   | Public Relations |
| Sonia   | Logistics       |
| Salman  | Public Relations |
| Rahim   | Logistics       |
| Karim   | Public Relations |
```

30) **Intersect ব্যবহার:**

```
SELECT name FROM Employee
INTERSECT
SELECT name FROM Department;
```

INTERSECT ব্যবহার করে কর্মচারীর এবং বিভাগের তালিকার মধ্যে যে কর্মচারীর এবং বিভাগের নাম মিলছে তাদের নাম দেখানো হয়েছে:

```
| name    |
|---------|
| Jamal   |
| Jeams   |
| Munni   |
| Sadia   |
| Sonia   |
| Salman  |
| Rahim   |
| Karim   |
```

```
| Rahim   | IT              |
| Karim   | Finance         |
| Munni   | NULL            |
| NULL    | Logistics       |
| NULL    | Public Relations |
```

28) **Limit ব্যবহার:**

```
SELECT name, salary FROM Employee LIMIT 3;
```

LIMIT ব্যবহার করে প্রথম 3 টি কর্মচারীর নাম এবং তাদের বেতন দেখানো হয়েছে:

```
| name    | salary |
|---------|--------|
| Jamal   | 50000  |
| Jeams   | 60000  |
| Munni   | 45000  |
```

31) **Except ব্যবহার:**

```
SELECT name FROM Employee
EXCEPT SELECT name FROM Department;
```

তালিকার মধ্যে যে কর্মচারীর এবং বিভাগের নাম মিলছে তাদের নাম বাদ দেখানো হয়েছে:

```
| name    |
|---------|
| Munni   |
| Salman  |
| Rahim   |
```

32) **ON ব্যবহার:**

```
SELECT Employee.name, Department.department_name
FROM Employee
JOIN Department ON Employee.department_id = Department.id;
```

ON ব্যবহার করে কর্মচারীর তালিকা এবং তাদের সাথে বিভাগের তালিকা মিলানো হয়েছে, তাদের মধ্যে যে কোন মিল থাকতে হবে

```
| name    | department_name |
|---------|-----------------|
| Jamal   | IT              |
```

33) **NOT EXISTS ব্যবহার:**

```
SELECT name
FROM Employee
WHERE NOT EXISTS (
  SELECT 1
  FROM Department
  WHERE Employee.department_id = Department.id
);
```

NOT EXISTS ব্যবহার করে তাদের যে কোন কর্মচারীর নাম দেখানো হয় যারা কোনো বিভাগে সংলগ্ন নয়:

```
| name    |
|---------|
| Munni   |
| Salman  |
| Rahim   |
```

```sql
 -- সমস্ত তথ্য দেখুন
SELECT * FROM students;
-- একটি নির্দিষ্ট কলামের তথ্য দেখুন
SELECT name, age FROM students;
-- শর্ত অনুযায়ী তথ্য দেখুন (উদাহরণস্বরূপ, যারা 18 বছরের কম)
SELECT * FROM students WHERE age < 18;
-- সর্ট করে তথ্য দেখুন
SELECT * FROM students ORDER BY age DESC;
-- নির্দিষ্ট সংখ্যক সারি দেখুন (উদাহরণস্বরূপ, প্রথম 5 সারি)
SELECT * FROM students LIMIT 5;
-- নির্দিষ্ট কলামের মাধ্যমে তথ্য যোগদান করুন
INSERT INTO students (name, age, grade, subject) VALUES
('Alice Johnson', 22, 'A', 'Science');
-- নির্দিষ্ট তথ্য আপডেট করুন (উদাহরণস্বরূপ, ছাত্র নম্বর 1-এর বয়স পরিবর্তন)
UPDATE students SET age = 20 WHERE id = 1;
-- নির্দিষ্ট তথ্য মোছুন (উদাহরণস্বরূপ, ছাত্র নম্বর 3)
DELETE FROM students WHERE id = 3;
-- দুই টেবিলের তথ্য যোগদান করুন (উদাহরণস্বরূপ, একটি নতুন টেবিল "courses" তৈরি করে এর সাথে সংযোজন)
CREATE TABLE courses (
    course_id INT PRIMARY KEY,
    course_name VARCHAR(255)
);
INSERT INTO courses (course_id, course_name) VALUES (101, 'Mathematics');
INSERT INTO courses (course_id, course_name) VALUES (102, 'English');
-- একটি নতুন কলাম তৈরি করুন (উদাহরণস্বরূপ, "semester" কলাম)
ALTER TABLE students ADD COLUMN semester INT;
-- কলামের তথ্য আপডেট করুন (উদাহরণস্বরূপ, ছাত্র নম্বর 2-এর "semester" তথ্য যোগ করুন)
UPDATE students SET semester = 2 WHERE id = 2;
-- একটি সংযোজন করে তথ্য দেখুন (উদাহরণস্বরূপ, ছাত্রের নাম এবং কোর্সের নাম)
SELECT students.name, courses.course_name
FROM students
```

```sql
-- একটি টেবিলের মধ্যে তথ্য দেখান এবং এর সাথে সাথে একটি নতুন কলাম তৈরি করুন (উদাহরণস্বরূপ, "students" টেবিলে)
SELECT * FROM students;
ALTER TABLE students ADD COLUMN new_column VARCHAR(255);
-- একটি কলামের মান দেখান এবং সেটি সরানো বা পরিবর্তন করা (উদাহরণস্বরূপ, "students" টেবিলে "new_column" কলামের জন্য)
SELECT new_column FROM students;
UPDATE students SET new_column = NULL;
-- একটি তালিকার তথ্য দেখান এবং সেটি সরানো বা পরিবর্তন করা (উদাহরণস্বরূপ, "new_table" তালিকা)
SELECT * FROM new_table;
DELETE FROM new_table;
-- কোন তালিকা তৈরি করুন এবং তার মধ্যে তথ্য ইনসার্ট করুন (উদাহরণস্বরূপ, "new_table")
CREATE TABLE new_table (
    id SERIAL PRIMARY KEY,
    description VARCHAR(255)
);
INSERT INTO new_table (description) VALUES ('Data 1'), ('Data 2'), ('Data 3');
 -- ডেটাবেসে সংবিদানমূলক তথ্য এবং তার বিশেষজ্ঞতা প্রদর্শন করুন (উদাহরণস্বরূপ, PostgreSQL)
SELECT current_database() AS database_name, current_user AS current_user, version();
-- একটি কলামের মানের তালিকা প্রদর্শন এবং সেটি মোছা বা পরিবর্তন করা (উদাহরণস্বরূপ, "students" টেবিলে "grade" কলামের জন্য)
SELECT DISTINCT grade FROM students;
UPDATE students SET grade = 'B' WHERE grade = 'C';
-- একটি কলামের সংখ্যার মধ্যে মান দেখান এবং সেটি একটি নতুন মানে পরিবর্তন করুন (উদাহরণস্বরূপ, "students" টেবিলে "age" কলামের জন্য)
SELECT MAX(age) FROM students;
UPDATE students SET age = 22 WHERE age = 21;
-- কোন কলামের মধ্যে কোন শব্দ অথবা অক্ষরের সাথে শুরু হওয়া তথ্য দেখুন (উদাহরণস্বরূপ, "students" টেবিলে "name" কলামের জন্য)
SELECT * FROM students WHERE name LIKE 'A%';
-- কোন কলামের মধ্যে কোন শব্দ অথবা অক্ষর থাকলে সংবিদানমূলক তথ্য দেখুন (উদাহরণস্বরূপ, "students" টেবিলে "name" কলামের জন্য)
```

```
INNER JOIN courses ON students.subject =
courses.course_name;
-- একটি সংযোজন করে এবং শর্ত অনুযায়ী তথ্য দেখুন (উদাহরণস্বরূপ, ছাত্রের নাম, বয়স এবং কোর্সের নাম, যারা 20 এর বেশি এবং ম্যাথেম্যাটিক্স কোর্স নিচ্ছে)
SELECT students.name, students.age, courses.course_name
FROM students
INNER JOIN courses ON students.subject =
courses.course_name
WHERE students.age > 20 AND courses.course_name =
'Mathematics';
-- কোন কলামে ডুপ্লিকেট মান থাকলে একবারে শুধুমাত্র একটি মাত্র তথ্য দেখুন (উদাহরণস্বরূপ, বয়স দ্বারা গ্রুপ করে)
SELECT age, COUNT(*) FROM students GROUP BY age HAVING
COUNT(*) > 1;
-- কতগুলি ছাত্র কোন কোর্স নিয়েছে তা গণনা করুন
SELECT subject, COUNT(*) as student_count FROM students
GROUP BY subject;
-- একটি কলামের যোগফল গণনা করুন (উদাহরণস্বরূপ, সবগুলি ছাত্রের বয়স)
SELECT SUM(age) FROM students;
-- নির্দিষ্ট শর্ত অনুযায়ী সংখ্যার কোন তথ্য থাকলে তা দেখুন, অন্যথায় ডিফল্ট মান দেখানোর জন্য COALESCE ব্যবহার করুন (উদাহরণস্বরূপ, কোন ছাত্রের "semester" নেই তবে 0 দেখানো)
SELECT name, COALESCE(semester, 0) FROM students;
-- কোন কলামের মধ্যে NULL মান থাকলে তা দেখুন (উদাহরণস্বরূপ, কারণ "semester" কলামে কিছু ছাত্রের জন্য তার মান নেই)
SELECT * FROM students WHERE semester IS NULL;
-- কোন কলামের মধ্যে নির্দিষ্ট মান না থাকলে তা দেখুন (উদাহরণস্বরূপ, কারণ "grade" কলামে কিছু ছাত্রের জন্য তার মান নেই)
SELECT * FROM students WHERE grade IS NOT NULL;
-- একটি সংখ্যার যোগফল বা গুণফল গণনা করুন (উদাহরণস্বরূপ, সবগুলি ছাত্রের বয়সের যোগফল)
SELECT SUM(age) FROM students;
 -- সংখ্যার ক্ষেত্রে ম্যাক্সিমাম এবং মিনিমাম মান দেখুন (উদাহরণস্বরূপ, বয়সের ম্যাক্সিমাম এবং মিনিমাম)
SELECT MAX(age) AS max_age, MIN(age) AS min_age FROM
students;
-- একটি তথ্যসেটের ক্ষেত্রে মধ্যকের মান দেখুন (উদাহরণস্বরূপ, ছাত্রের বয়সের মধ্যকের মান)
SELECT PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY age) AS
median_age FROM students;
-- একটি কলামের মানের যোগফল দেখুন (উদাহরণস্বরূপ, সবগুলি ছাত্রের বয়সের যোগফল)
SELECT SUM(age) FROM students;
-- কোন কলামের সবগুলি মানের গড় দেখুন (উদাহরণস্বরূপ, ছাত্রের সট করা বয়সের গড়)
SELECT AVG(age) FROM students;
```

```
SELECT * FROM students WHERE name ILIKE '%john%';
-- একটি তালিকার তথ্য দেখুন এবং এর মধ্যে যেগুলি দুটি তালিকার মধ্যে সাম্য আছে তা দেখান (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
SELECT * FROM students
INTERSECT
SELECT * FROM new_table;
-- একটি তালিকা থেকে ডেটা মুছুন যা অন্য একটি তালিকার মধ্যে নেই (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
SELECT * FROM students
EXCEPT
SELECT * FROM new_table;
-- একটি তালিকা তৈরি করুন এবং তার মধ্যে যেগুলি দুটি তালিকার মধ্যে কোন সাম্য নেই তা দেখান (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
SELECT * FROM students
FULL OUTER JOIN new_table ON students.name = new_table.description
WHERE students.name IS NULL OR new_table.description IS NULL;
-- একটি তালিকা এবং একটি সাবকুয়েরি ব্যবহার করে তথ্য দেখুন (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
SELECT * FROM students
WHERE age > (SELECT AVG(age) FROM students);
-- একটি তালিকা তৈরি করুন এবং তার মধ্যে যেগুলি একটি অন্য তালিকার সাথে সম্পর্কিত নেই তা দেখান (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
CREATE TABLE non_matching_students AS
SELECT * FROM students
WHERE NOT EXISTS (
    SELECT 1 FROM new_table
    WHERE students.name = new_table.description
);
[17/11/2023 01:48] Myself(016): -- একটি তালিকা তৈরি করুন এবং তার মধ্যে তথ্য দেখুন এবং সেটি সারানো (উদাহরণস্বরূপ, "new_table")
CREATE TABLE new_table (
    id SERIAL PRIMARY KEY,
    description VARCHAR(255)
);
INSERT INTO new_table (description) VALUES ('Data 1'), ('Data 2'),
('Data 3');
-- একটি তালিকা তৈরি করুন এবং একটি অন্য তালিকার ডেটা থেকে তার সাথে সংযোগ করুন (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
CREATE TABLE joined_data AS
SELECT students.*, new_table.description
FROM students
JOIN new_table ON students.id = new_table.id;
-- একটি সার্চ কোয়েরি চালানো এবং তার ফলাফল দেখান (উদাহরণস্বরূপ, "students" তালিকা)
SELECT * FROM students WHERE to_tsvector('english', name) @@
to_tsquery('english', 'John');
-- ডেটাবেসের বিভিন্ন তালিকার মধ্যে যেগুলি একে অপরের সাথে সম্পর্কিত তা দেখুন (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
```

```sql
-- কোন কলামের মানের সংখ্যার গড় দেখুন (উদাহরণস্বরূপ, সবগুলি ছাত্রের বয়সের গড়)
SELECT AVG(CAST(age AS DECIMAL)) FROM students;
-- একটি কলামের মানের যোগফল এবং তার সংখ্যা দেখুন (উদাহরণস্বরূপ, সবগুলি ছাত্রের বয়সের যোগফল এবং তাদের সংখ্যা)
SELECT SUM(age) AS total_age, COUNT(*) AS total_students
FROM students;
-- সংখ্যা মূল্যের ক্ষেত্রে বৃদ্ধি এবং মিনিমাম দেখুন (উদাহরণস্বরূপ, বৃদ্ধি এবং মিনিমাম বয়স)
SELECT PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY age)
AS age_75th_percentile,
       PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY age)
AS age_25th_percentile
FROM students;
-- দুই টেবিলের তথ্য যোগদান এবং সম্পাদনা করতে একটি INNER JOIN ব্যবহার করুন (উদাহরণস্বরূপ, ছাত্রের নাম, বয়স এবং কোর্সের নাম)
SELECT students.name, students.age, courses.course_name
FROM students
INNER JOIN courses ON students.subject =
courses.course_name;
-- তিন টেবিলের তথ্য যোগদান এবং সম্পাদনা করতে একটি INNER JOIN এবং LEFT JOIN ব্যবহার করুন (উদাহরণস্বরূপ, ছাত্রের নাম, কোর্সের নাম এবং তাদের জন্য উপলব্ধ কোর্সের নাম)
SELECT students.name, COALESCE(courses.course_name, 'Not
Enrolled') AS enrolled_course,
additional_courses.course_name AS additional_course
FROM students
LEFT JOIN courses ON students.subject =
courses.course_name
LEFT JOIN additional_courses ON students.id =
additional_courses.student_id;
-- কোন কলামের মান গড় বের করুন এবং এর সাথে সাথে প্রতি সারির বয়স থেকে বিয়োগ করুন (উদাহরণস্বরূপ, প্রতি ছাত্রের বয়স থেকে গড় বয়স বিয়োগ করুন)
SELECT age - AVG(age) OVER () AS age_difference FROM
students;
-- প্রতি ছাত্রের জন্য শ্রেণীর পুরানো মান এবং নতুন মান দেখুন (উদাহরণস্বরূপ, প্রতি ছাত্রের জন্য বয়স পরিবর্তনের জন্য LAG এবং LEAD ব্যবহার করুন)
SELECT name, age,
       LAG(age) OVER (ORDER BY age) AS previous_age,
       LEAD(age) OVER (ORDER BY age) AS next_age
FROM students;
-- ডেটাবেস প্রদর্শন করতে INFORMATION_SCHEMA ব্যবহার করুন (উদাহরণস্বরূপ, সব টেবিলের তালিকা)
SELECT table_name
FROM information_schema.tables
WHERE table_schema = 'public';
```

```sql
SELECT conname, conrelid::regclass AS table_from,
confrelid::regclass AS table_to
FROM pg_constraint
WHERE confrelid = 'students'::regclass;
-- কোন সংখ্যার সিকুয়েন্স বা কয়েরি ব্যবহার করে ডেটাবেসে নতুন তথ্য যোগ করুন (উদাহরণস্বরূপ, নতুন ছাত্র)
INSERT INTO students (id, name, age, grade, subject) VALUES
(DEFAULT, 'New Student', 21, 'A', 'Physics');
-- একটি সংখ্যার সিকুয়েন্স বা কয়েরি থেকে নতুন নাম্বার তৈরি করুন এবং এর সাথে সাথে একটি নতুন তথ্য যোগ করুন (উদাহরণস্বরূপ, নতুন ছাত্র)
INSERT INTO students (id, name, age, grade, subject) VALUES
(NEXTVAL('student_id_sequence'), 'New Student 2', 22, 'B',
'Chemistry');
-- একটি সংখ্যার সিকুয়েন্স বা কয়েরি ব্যবহার করে ডেটাবেস থেকে তথ্য সরানো (উদাহরণস্বরূপ, শুধুমাত্র প্রথম ছাত্র)
DELETE FROM students WHERE id = 1001;
-- সংখ্যার এ
[17/11/2023 01:49] Myself(016): -- সংখ্যার একটি সিকুয়েন্স বা কয়েরি থেকে নতুন নাম্বার তৈরি করুন এবং সেটি সাথে সাথে একটি অতিরিক্ত কলামের তথ্য যোগ করুন (উদাহরণস্বরূপ, ছাত্রের জন্য অতিরিক্ত কোর্স)
INSERT INTO students (id, name, age, grade, subject, semester)
VALUES (NEXTVAL('student_id_sequence'), 'Additional Student', 23,
'A', 'History', 2);
-- কোন কলামের সংখ্যার মধ্যে একটি রেঞ্জের মধ্যে পড়া তথ্য দেখুন (উদাহরণস্বরূপ, বয়স 20 থেকে 25)
SELECT * FROM students WHERE age BETWEEN 20 AND 25;
-- একটি কলামের মানের সাথে সাথে একটি অপারেটর বা ফাংশন ব্যবহার করে তথ্য দেখুন (উদাহরণস্বরূপ, বয়সের সংখ্যার স্কোয়ার রুট)
SELECT name, age, SQRT(age) AS sqrt_age FROM students;
-- প্রতি ছাত্রের প্রতি কোর্সে একটি মান এবং তার সাথে সাথে তার প্রতি সারি হিসেবে একটি কলাম যোগ করুন (উদাহরণস্বরূপ, বয়স এবং কোর্সের মাধ্যমে)
SELECT *,
       AVG(age) OVER (PARTITION BY subject) AS avg_age_per_course
FROM students;
-- ডেটাবেসের বিভিন্ন তালিকার মধ্যে সংখ্যা এবং তাদের সংখ্যা দেখান
SELECT table_name, COUNT(*) AS row_count
FROM information_schema.tables
GROUP BY table_name
ORDER BY row_count DESC;
-- একটি কলামের মানের সাথে সাথে তার প্রতি সারি হিসেবে একটি কলাম যোগ করুন (উদাহরণস্বরূপ, বয়সের সংখ্যার মোট)
SELECT *, SUM(age) OVER () AS total_age FROM students;
[17/11/2023 01:51] Myself(016): -- একটি তালিকা তৈরি করুন এবং তার মধ্যে যেগুলি একটি অন্য তালিকার সাথে সম্পর্কিত নেই তা দেখান (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
CREATE TABLE non_matching_students AS
SELECT * FROM students
WHERE NOT EXISTS (
```

```sql
-- একটি কলামের মানের ডেসকেন্ডিং ক্রমে শোর্ট করুন (উদাহরণস্বরূপ, বয়সের ক্রমে ছাত্রদের তালিকা)
SELECT * FROM students ORDER BY age DESC;
-- সংখ্যার একটি সিকুয়েন্স বা কয়েরি সৃষ্টি করুন (উদাহরণস্বরূপ, একটি ছাত্রের নতুন নাম্বার যোগ করুন)
CREATE SEQUENCE student_id_sequence START 1001;
-- সংখ্যার একটি সিকুয়েন্স বা কয়েরি ব্যবহার করে ডেটাবেসে নতুন তথ্য যোগ করুন (উদাহরণস্বরূপ, নতুন ছাত্র)
INSERT INTO students (id, name, age, grade, subject)
VALUES (NEXTVAL('student_id_sequence'), 'New Student', 21,
'A', 'Physics');
-- সংখ্যার একটি সিকুয়েন্স থেকে নতুন নাম্বার তৈরি করুন (উদাহরণস্বরূপ, নতুন কোর্স যোগ করার জন্য)
INSERT INTO courses (course_id, course_name) VALUES
(NEXTVAL('course_id_sequence'), 'Computer Science');
-- একটি সংখ্যার সিকুয়েন্স পুনরায় সৃষ্টি করুন (উদাহরণস্বরূপ, একই ধরনের তালিকা তৈরি করতে)
CREATE SEQUENCE additional_course_id_sequence START 201;
-- সংখ্যার একটি সিকুয়েন্স থেকে নতুন নাম্বার তৈরি করুন এবং এর সাথে সাথে একটি অতিরিক্ত কলামের তথ্য যোগ করুন (উদাহরণস্বরূপ, ছাত্রের জন্য অতিরিক্ত কোর্স)
INSERT INTO additional_courses (id, student_id,
course_name) VALUES
(NEXTVAL('additional_course_id_sequence'), 1,
'Statistics');
-- একটি কলামের মানের মধ্যে কোন শব্দ অথবা অক্ষরের সাথে শুরু হওয়া সংখ্যাগুলি দেখুন (উদাহরণস্বরূপ, সবচেয়ে ছোট এবং বড় বয়স)
SELECT * FROM students WHERE age LIKE '2%' OR age LIKE
'1%';
-- কোন সংখ্যার একটি রেঞ্জের মধ্যে পড়া তথ্য দেখুন (উদাহরণস্বরূপ, বয়স 18 থেকে 22)
SELECT * FROM students WHERE age BETWEEN 18 AND 22;
-- একটি কলামের সংখ্যার সাথে সাথে একটি অপারেটর বা ফাংশন ব্যবহার করে তথ্য দেখুন (উদাহরণস্বরূপ, বয়সের সংখ্যার স্কোয়ার রুট)
SELECT name, age, SQRT(age) AS sqrt_age FROM students;
-- ডেটাবেস ব্যবহারকারীর জন্য অনুমতি চেক করুন (উদাহরণস্বরূপ, ডেটাবেস অ্যাডমিন)
SHOW GRANTS FOR admin_user;
-- টেবিল স্কিমা দেখুন (উদাহরণস্বরূপ, "students" টেবিলের স্কিমা)
DESCRIBE students;
-- একটি ইনডেক্স তৈরি করুন (উদাহরণস্বরূপ, "students" টেবিলে "name" কলামের জন্য)
CREATE INDEX idx_students_name ON students(name);
-- কোন কলামের সংখ্যার মধ্যে একটি স্বীকৃত মান থাকলে তার জন্য তথ্য দেখুন (উদাহরণস্বরূপ, "grade" কলামের জন্য 'A' স্বীকৃতি)
SELECT * FROM students WHERE grade = 'A';
-- প্রতি ছাত্রের প্রতি কোর্সে একটি মান এবং তার সাথে সাথে তার প্রতি সারি হিসেবে একটি কলাম যোগ করুন (উদাহরণস্বরূপ, বয়স এবং কোর্সের মাধ্যমে)
    SELECT 1 FROM new_table
    WHERE students.name = new_table.description
);
-- একটি তালিকা তৈরি করুন এবং তার মধ্যে যেগুলি দুটি তালিকার মধ্যে সাম্য আছে তা দেখান (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
CREATE TABLE matching_students AS
SELECT * FROM students
INTERSECT
SELECT * FROM new_table;
-- একটি তালিকা তৈরি করুন এবং একটি অন্য তালিকার ডেটা থেকে তার সাথে সংযোগ করুন (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
CREATE TABLE joined_data AS
SELECT students.*, new_table.description
FROM students
JOIN new_table ON students.id = new_table.id;
-- একটি তালিকা থেকে ডেটা মুছুন যা অন্য একটি তালিকার মধ্যে নেই (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
SELECT * FROM students
EXCEPT
SELECT * FROM new_table;
-- কোন তালিকা তৈরি করুন এবং তার মধ্যে তথ্য ইনসার্ট করুন (উদাহরণস্বরূপ, "new_table")
CREATE TABLE new_table (
    id SERIAL PRIMARY KEY,
    description VARCHAR(255)
);
INSERT INTO new_table (description) VALUES ('Data 1'), ('Data 2'),
('Data 3');
[17/11/2023 01:51] Myself(016): -- একটি তালিকা তৈরি করুন এবং তার মধ্যে যেগুলি একটি অন্য তালিকার সাথে সম্পর্কিত নেই তা দেখান (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
CREATE TABLE non_matching_students AS
SELECT * FROM students
WHERE NOT EXISTS (
    SELECT 1 FROM new_table
    WHERE students.name = new_table.description
);
-- একটি তালিকা তৈরি করুন এবং তার মধ্যে যেগুলি দুটি তালিকার মধ্যে সাম্য আছে তা দেখান (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
CREATE TABLE matching_students AS
SELECT * FROM students
INTERSECT
SELECT * FROM new_table;
-- একটি তালিকা তৈরি করুন এবং একটি অন্য তালিকার ডেটা থেকে তার সাথে সংযোগ করুন (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
CREATE TABLE joined_data AS
SELECT students.*, new_table.description
FROM students
```

```
SELECT *,
       AVG(age) OVER (PARTITION BY subject) AS
avg_age_per_course
FROM students;
-- ডেটাবেসের বিভিন্ন তালিকার ম
 -- ডেটাবেসের বিভিন্ন তালিকার মধ্যে একটি সংখ্যা এবং তাদের সংখ্যা দেখুন
SELECT table_name, COUNT(*) AS row_count
FROM information_schema.tables
GROUP BY table_name
ORDER BY row_count DESC;
-- কোন কলামের জন্য একটি ইনডেক্স থাকলে তা দেখুন (উদাহরণস্বরূপ,
"students" টেবিলে "name" কলামের জন্য)
SHOW INDEX FROM students WHERE column_name = 'name';
-- একটি তালিকা তৈরি করুন এবং তার মধ্যে ডেটা ইনসার্ট করুন
(উদাহরণস্বরূপ, "new_table")
CREATE TABLE new_table (
    id SERIAL PRIMARY KEY,
    description VARCHAR(255)
);
INSERT INTO new_table (description) VALUES ('Data 1'),
('Data 2'), ('Data 3');
-- ডেটাবেস ব্যবহারকারীর জন্য একটি নতুন ব্যবহারকারী তৈরি করুন এবং
অনুমতি দান (উদাহরণস্বরূপ, "new_user")
CREATE USER new_user WITH PASSWORD 'password';
GRANT ALL PRIVILEGES ON DATABASE your_database TO
new_user;
-- একটি কলামের মান আপডেট করুন যেগুলির মধ্যে কোন কলামের মান
NULL আছে (উদাহরণস্বরূপ, "semester" কলামের জন্য)
UPDATE students SET semester = 1 WHERE semester IS NULL;
-- ডেটাবেসের বিভিন্ন তালিকা বা তালিকার কলামের নাম দেখুন
SELECT table_name, column_name
FROM information_schema.columns
WHERE table_schema = 'public';
-- কোন কলামের মধ্যে কোন নির্দিষ্ট মান থাকলে সম্পর্কিত সারি দেখুন
(উদাহরণস্বরূপ, "students" টেবিলে "subject" কলামের জন্য "Math"
মান)
SELECT * FROM students WHERE subject = 'Math';
-- একটি তালিকা থেকে ডেটা মুছুন (উদাহরণস্বরূপ, "new_table" তালিকা)
DELETE FROM new_table;
-- ডেটাবেস থেকে একটি তালিকা সরানো (উদাহরণস্বরূপ, "new_table")
DROP TABLE IF EXISTS new_table;
 -- একটি কলামের মান সংক্ষেপে একটি সারি বা স্তম্ভ দেখান (উদাহরণস্বরূপ,
"students" টেবিলে "name" কলামের জন্য)
SELECT DISTINCT name FROM students;
-- একটি টেবিলের তালিকা দেখান (উদাহরণস্বরূপ, "students" টেবিল)
SHOW TABLES;
-- একটি টেবিলের তথ্য দেখান (উদাহরণস্বরূপ, "students" টেবিল)
SELECT * FROM students;
```

```
JOIN new_table ON students.id = new_table.id;
-- একটি তালিকা থেকে ডেটা মুছুন যা অন্য একটি তালিকার মধ্যে নেই (উদাহরণস্বরূপ,
"students" এবং "new_table" তালিকা)
SELECT * FROM students
EXCEPT
SELECT * FROM new_table;
-- কোন তালিকা তৈরি করুন এবং তার মধ্যে তথ্য ইনসার্ট করুন (উদাহরণস্বরূপ,
"new_table")
CREATE TABLE new_table (
    id SERIAL PRIMARY KEY,
    description VARCHAR(255)
);
INSERT INTO new_table (description) VALUES ('Data 1'), ('Data 2'),
('Data 3');
[17/11/2023 01:57] Myself(016): -- ডেটাবেসে সংবিদানমূলক তথ্য এবং তার
বিশেষজ্ঞতা প্রদর্শন করুন (উদাহরণস্বরূপ, PostgreSQL)
SELECT current_database() AS database_name, current_user AS
current_user, version();
-- একটি কলামের মানের তালিকা প্রদর্শন এবং সেটি মোছা বা পরিবর্তন করা
(উদাহরণস্বরূপ, "students" টেবিলে "grade" কলামের জন্য)
SELECT DISTINCT grade FROM students;
UPDATE students SET grade = 'B' WHERE grade = 'C';
-- একটি কলামের সংখ্যার মধ্যে মান দেখান এবং সেটি একটি নতুন মানে পরিবর্তন করুন
(উদাহরণস্বরূপ, "students" টেবিলে "age" কলামের জন্য)
SELECT MAX(age) FROM students;
UPDATE students SET age = 22 WHERE age = 21;
-- কোন কলামের মধ্যে কোন শব্দ অথবা অক্ষরের সাথে শুরু হওয়া তথ্য দেখুন
(উদাহরণস্বরূপ, "students" টেবিলে "name" কলামের জন্য)
SELECT * FROM students WHERE name LIKE 'A%';
-- কোন কলামের মধ্যে কোন শব্দ অথবা অক্ষর থাকলে সংবিদানমূলক তথ্য দেখুন
(উদাহরণস্বরূপ, "students" টেবিলে "name" কলামের জন্য)
SELECT * FROM students WHERE name ILIKE '%john%';
-- একটি তালিকার তথ্য দেখুন এবং এর মধ্যে যেগুলি দুটি তালিকার মধ্যে সাম্য আছে তা
দেখান (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
SELECT * FROM students
INTERSECT
SELECT * FROM new_table;
-- একটি তালিকা থেকে ডেটা মুছুন যা অন্য একটি তালিকার মধ্যে নেই (উদাহরণস্বরূপ,
"students" এবং "new_table" তালিকা)
SELECT * FROM students
EXCEPT
SELECT * FROM new_table;
-- একটি তালিকা তৈরি করুন এবং তার মধ্যে যেগুলি দুটি তালিকার মধ্যে কোন সাম্য নেই
তা দেখান (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
SELECT * FROM students
FULL OUTER JOIN new_table ON students.name = new_table.description
WHERE students.name IS NULL OR new_table.description IS NULL;
```

```sql
-- একটি টেবিলের নির্দিষ্ট কলামের মধ্যে মোট মানের সংখ্যা দেখান (উদাহরণস্বরূপ, "students" টেবিলে "age" কলামের জন্য)
SELECT COUNT(age) FROM students;
-- একটি টেবিলের নির্দিষ্ট কলামের মাধ্যম মান দেখান (উদাহরণস্বরূপ, "students" টেবিলে "age" কলামের জন্য)
SELECT AVG(age) FROM students;
-- একটি টেবিলের নির্দিষ্ট কলামের মাধ্যমে সর্ট করে প্রথম কয়েটি সারি দেখান (উদাহরণস্বরূপ, "students" টেবিলে "age" কলামের জন্য)
SELECT * FROM students ORDER BY age LIMIT 5;
-- একটি স্কিমা তৈরি করুন এবং তার মধ্যে তথ্য দেখান (উদাহরণস্বরূপ, "new_schema")
CREATE SCHEMA new_schema;
```

```sql
-- একটি তালিকা এবং একটি সাবকুয়েরি ব্যবহার করে তথ্য দেখুন (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
SELECT * FROM students
WHERE age > (SELECT AVG(age) FROM students);
-- একটি তালিকা তৈরি করুন এবং তার মধ্যে যেগুলি একটি অন্য তালিকার সাথে সম্পর্কিত নেই তা দেখান (উদাহরণস্বরূপ, "students" এবং "new_table" তালিকা)
CREATE TABLE non_matching_students AS
SELECT * FROM students
WHERE NOT EXISTS (
    SELECT 1 FROM new_table
    WHERE students.name = new_table.description
);
```