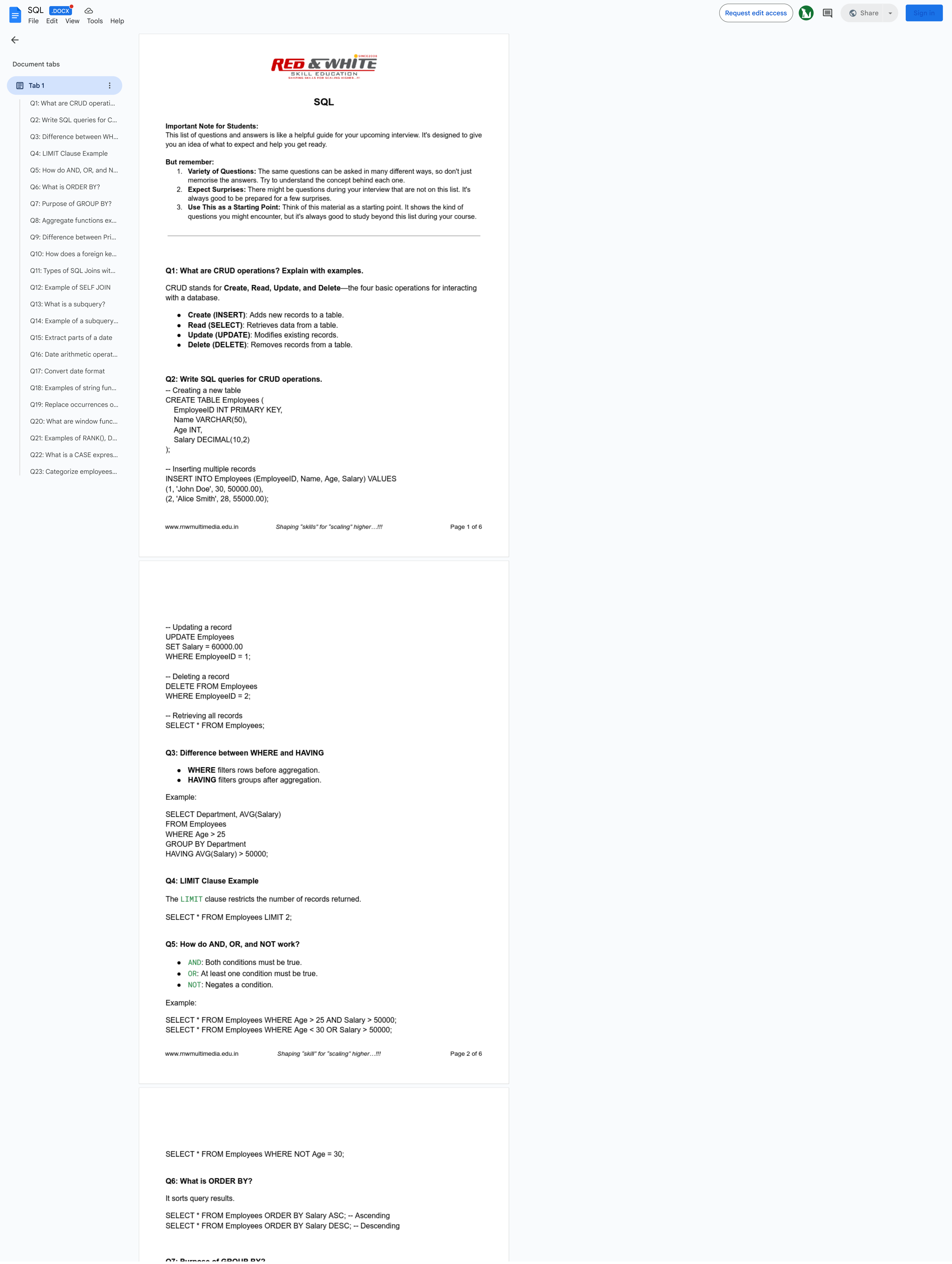
SQL .DOCX
File Edit View Tools Help
Request edit access
Share
Sign in
Document tabs
Tab 1
Q1: What are CRUD operati...
Q2: Write SQL queries for C...
Q3: Difference between WH...
Q4: LIMIT Clause Example
Q5: How do AND, OR, and N...
Q6: What is ORDER BY?
Q7: Purpose of GROUP BY?
Q8: Aggregate functions ex...
Q9: Difference between Pri...
Q10: How does a foreign ke...
Q11: Types of SQL Joins wit...
Q12: Example of SELF JOIN
Q13: What is a subquery?
Q14: Example of a subquery...
Q15: Extract parts of a date
Q16: Date arithmetic operat...
Q17: Convert date format
Q18: Examples of string fun...
Q19: Replace occurrences o...
Q20: What are window func...
Q21: Examples of RANK(), D...
Q22: What is a CASE expres...
Q23: Categorize employees...

RED & WHITE
SKILL EDUCATION

# SQL

**Important Note for Students:**
This list of questions and answers is like a helpful guide for your upcoming interview. It's designed to give you an idea of what to expect and help you get ready.

**But remember:**

1. **Variety of Questions:** The same questions can be asked in many different ways, so don't just memorise the answers. Try to understand the concept behind each one.
2. **Expect Surprises:** There might be questions during your interview that are not on this list. It's always good to be prepared for a few surprises.
3. **Use This as a Starting Point:** Think of this material as a starting point. It shows the kind of questions you might encounter, but it's always good to study beyond this list during your course.

---

### Q1: What are CRUD operations? Explain with examples.

CRUD stands for **Create, Read, Update, and Delete**—the four basic operations for interacting with a database.

- **Create (INSERT)**: Adds new records to a table.
- **Read (SELECT)**: Retrieves data from a table.
- **Update (UPDATE)**: Modifies existing records.
- **Delete (DELETE)**: Removes records from a table.

### Q2: Write SQL queries for CRUD operations.

```
-- Creating a new table
CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   Name VARCHAR(50),
   Age INT,
   Salary DECIMAL(10,2)
);

-- Inserting multiple records
INSERT INTO Employees (EmployeeID, Name, Age, Salary) VALUES
(1, 'John Doe', 30, 50000.00),
(2, 'Alice Smith', 28, 55000.00);
```

```
-- Updating a record
UPDATE Employees
SET Salary = 60000.00
WHERE EmployeeID = 1;

-- Deleting a record
DELETE FROM Employees
WHERE EmployeeID = 2;

-- Retrieving all records
SELECT * FROM Employees;
```

### Q3: Difference between WHERE and HAVING

- **WHERE** filters rows before aggregation.
- **HAVING** filters groups after aggregation.

Example:

```
SELECT Department, AVG(Salary)
FROM Employees
WHERE Age > 25
GROUP BY Department
HAVING AVG(Salary) > 50000;
```

### Q4: LIMIT Clause Example

The `LIMIT` clause restricts the number of records returned.

```
SELECT * FROM Employees LIMIT 2;
```

### Q5: How do AND, OR, and NOT work?

- `AND`: Both conditions must be true.
- `OR`: At least one condition must be true.
- `NOT`: Negates a condition.

Example:

```
SELECT * FROM Employees WHERE Age > 25 AND Salary > 50000;
SELECT * FROM Employees WHERE Age < 30 OR Salary > 50000;
```

```
SELECT * FROM Employees WHERE NOT Age = 30;
```

### Q6: What is ORDER BY?

It sorts query results.

```
SELECT * FROM Employees ORDER BY Salary ASC; -- Ascending
SELECT * FROM Employees ORDER BY Salary DESC; -- Descending
```

### Q7: Purpose of GROUP BY?

**Q7: Purpose of GROUP BY?**

It groups rows with the same values and applies aggregate functions.

```
SELECT Department, COUNT(*) FROM Employees GROUP BY Department;
```

**Q8: Aggregate functions examples**

```
SELECT SUM(Salary) FROM Employees;
SELECT AVG(Salary) FROM Employees;
SELECT MAX(Salary) FROM Employees;
SELECT MIN(Salary) FROM Employees;
SELECT COUNT(*) FROM Employees;
```

**Q9: Difference between Primary Key and Foreign Key**

- **Primary Key** uniquely identifies a row in a table.
- **Foreign Key** establishes a relationship between two tables.

**Q10: How does a foreign key maintain referential integrity?**

A **foreign key** ensures that a referenced value exists in the parent table.

```
CREATE TABLE Departments (
   DeptID INT PRIMARY KEY,
   DeptName VARCHAR(50)
);

CREATE TABLE Employees (
```

```
   EmployeeID INT PRIMARY KEY,
   Name VARCHAR(50),
   DeptID INT,
   FOREIGN KEY (DeptID) REFERENCES Departments(DeptID)
);
```

**Q11: Types of SQL Joins with examples**

```
-- INNER JOIN (Common records in both tables)
SELECT Employees.Name, Departments.DeptName
FROM Employees
INNER JOIN Departments ON Employees.DeptID = Departments.DeptID;

-- LEFT JOIN (All records from left, matched from right)
SELECT Employees.Name, Departments.DeptName
FROM Employees
LEFT JOIN Departments ON Employees.DeptID = Departments.DeptID;

-- RIGHT JOIN (All records from right, matched from left)
SELECT Employees.Name, Departments.DeptName
FROM Employees
RIGHT JOIN Departments ON Employees.DeptID = Departments.DeptID;

-- FULL JOIN (All records from both tables)
SELECT Employees.Name, Departments.DeptName
FROM Employees
FULL JOIN Departments ON Employees.DeptID = Departments.DeptID;
```

**Q12: Example of SELF JOIN**

```
SELECT e1.Name AS Employee, e2.Name AS Manager
FROM Employees e1
JOIN Employees e2 ON e1.ManagerID = e2.EmployeeID;
```

**Q13: What is a subquery?**

A subquery is a query inside another query.

**Q14: Example of a subquery for salary comparison**

```
SELECT Name, Salary FROM Employees
WHERE Salary > (SELECT AVG(Salary) FROM Employees);
```

**Q15: Extract parts of a date**

```
SELECT YEAR(CURDATE()), MONTH(CURDATE()), DAY(CURDATE());
```

**Q16: Date arithmetic operations**

```
SELECT DATE_ADD(CURDATE(), INTERVAL 7 DAY); -- Adds 7 days
SELECT DATEDIFF('2024-12-31', CURDATE()); -- Days between two dates
```

**Q17: Convert date format**

```
SELECT DATE_FORMAT(CURDATE(), '%d-%m-%Y');
```

**Q18: Examples of string functions**

```
SELECT CONCAT('Hello', ' World');
SELECT SUBSTRING('Database', 1, 4);
SELECT REPLACE('Database', 'Data', 'Info');
SELECT UPPER('database');
SELECT TRIM('  Hello  ');
```

**Q19: Replace occurrences of "old" with "new"**

```
SELECT REPLACE(Description, 'old', 'new') FROM Products;
```

**Q20: What are window functions?**

Window functions operate over a subset of rows, unlike aggregate functions.

Window functions operate over a subset of rows, unlike aggregate functions.

**Q21: Examples of `RANK()`, `DENSE_RANK()`, and `ROW_NUMBER()`**

```
SELECT Name, Salary, RANK() OVER (ORDER BY Salary DESC) AS Rank FROM
Employees;
SELECT Name, Salary, DENSE_RANK() OVER (ORDER BY Salary DESC) AS DenseRank
FROM Employees;
```

```
SELECT Name, Salary, ROW_NUMBER() OVER (ORDER BY Salary DESC) AS RowNum
FROM Employees;
```

**Q22: What is a CASE expression?**

The `CASE` expression provides conditional logic inside SQL queries.

**Q23: Categorize employees based on salary**

```
SELECT Name, Salary,
   CASE
      WHEN Salary > 70000 THEN 'High'
      WHEN Salary BETWEEN 50000 AND 70000 THEN 'Medium'
      ELSE 'Low'
   END AS SalaryCategory
FROM Employees;
```