SQL Student Notes: Clauses, Operators, and Functions

Table of Contents

SQL Clauses

SQL clauses are keywords that define different parts of a SQL statement. Think of them as building blocks that structure your query.

Common SQL Clauses

1. SELECT Clause

Purpose: Specifies which columns to retrieve from the database

```sql
SELECT column1, column2 FROM table_name;
```

2. FROM Clause

Purpose: Specifies the table(s) to query data from

```sql
SELECT * FROM customers;
```

3. WHERE Clause

Purpose: Filters records based on specified conditions

```sql
SELECT * FROM accounts WHERE balance > 1000;
```

4. ORDER BY Clause

Purpose: Sorts the result set in ascending or descending order

```sql
```

```sql
SELECT * FROM transactions ORDER BY amount DESC;
```

5. GROUP BY Clause

Purpose: Groups rows that have the same values in specified columns

```
sql

SELECT customer_id, COUNT(*) FROM transactions GROUP BY customer_id;
```

6. HAVING Clause

Purpose: Filters groups created by GROUP BY (like WHERE but for groups)

```
sql

SELECT customer_id, SUM(amount) FROM transactions
GROUP BY customer_id HAVING SUM(amount) > 5000;
```

Financial Domain Example

Let's say we have a **bank_accounts** table:

account_id	customer_name	account_type	balance	branch_id
1001	John Smith	Savings	2500.00	101
1002	Jane Doe	Checking	1200.00	102
1003	Bob Johnson	Savings	5000.00	101

Query: Find all savings accounts with balance greater than $2000, ordered by balance

```
sql

SELECT account_id, customer_name, balance
FROM bank_accounts
WHERE account_type = 'Savings' AND balance > 2000
ORDER BY balance DESC;
```

SQL Operators

SQL operators are symbols or keywords used to perform operations on data. They help you create conditions and perform calculations.

Types of SQL Operators

1. Arithmetic Operators

Used for mathematical calculations

Operator	Description	Example
+	Addition	`SELECT balance + 100`
-	Subtraction	`SELECT balance - 50`
*	Multiplication	`SELECT balance * 1.05`
/	Division	`SELECT balance / 2`
%	Modulo	`SELECT account_id % 2`

2. Comparison Operators

Used to compare values

Operator	Description	Example
=	Equal to	`WHERE balance = 1000`
!= or <>	Not equal to	`WHERE balance != 0`
>	Greater than	`WHERE balance > 500`
<	Less than	`WHERE balance < 1000`
>=	Greater than or equal	`WHERE balance >= 1000`
<=	Less than or equal	`WHERE balance <= 5000`

3. Logical Operators

Used to combine or modify conditions

Operator	Description	Example
AND	Both conditions must be true	`WHERE balance > 1000 AND account_type = 'Savings'`
OR	At least one condition must be true	`WHERE balance > 5000 OR account_type = 'Premium'`
NOT	Negates a condition	`WHERE NOT account_type = 'Closed'`

4. Pattern Matching Operators

Used for text pattern matching

Operator	Description	Example
LIKE	Pattern matching with wildcards	`WHERE customer_name LIKE 'John%'`
IN	Matches any value in a list	`WHERE branch_id IN (101, 102, 103)`
BETWEEN	Checks if value is within a range	`WHERE balance BETWEEN 1000 AND 5000`

Financial Domain Example

Query: Find all checking or savings accounts with balance between $1000 and $10000

```sql
SELECT account_id, customer_name, balance, account_type
FROM bank_accounts
WHERE account_type IN ('Checking', 'Savings')
AND balance BETWEEN 1000 AND 10000
ORDER BY balance DESC;
```

SQL Functions

SQL functions are built-in programs that perform specific operations on data. They can manipulate, calculate, or transform data.

Types of SQL Functions

1. Aggregate Functions

Perform calculations on multiple rows and return a single value

Function	Description	Example
COUNT()	Counts number of rows	`SELECT COUNT(*) FROM transactions`
SUM()	Calculates total sum	`SELECT SUM(amount) FROM transactions`
AVG()	Calculates average	`SELECT AVG(balance) FROM accounts`
MAX()	Finds maximum value	`SELECT MAX(balance) FROM accounts`
MIN()	Finds minimum value	`SELECT MIN(balance) FROM accounts`

2. String Functions

Manipulate text data

Function	Description	Example
UPPER()	Converts to uppercase	`SELECT UPPER(customer_name)`
LOWER()	Converts to lowercase	`SELECT LOWER(customer_name)`
LENGTH()	Returns string length	`SELECT LENGTH(customer_name)`
SUBSTRING()	Extracts part of string	`SELECT SUBSTRING(account_id, 1, 3)`

3. Date Functions

Work with date and time data

Function	Description	Example
NOW()	Current date and time	`SELECT NOW()`
YEAR()	Extracts year from date	`SELECT YEAR(transaction_date)`
MONTH()	Extracts month from date	`SELECT MONTH(transaction_date)`
DATEDIFF()	Difference between dates	`SELECT DATEDIFF(NOW(), account_opened)`

4. Mathematical Functions

Perform mathematical operations

Function	Description	Example
ROUND()	Rounds to specified decimals	`SELECT ROUND(balance, 2)`
ABS()	Absolute value	`SELECT ABS(transaction_amount)`
CEILING()	Rounds up	`SELECT CEILING(balance/1000)`
FLOOR()	Rounds down	`SELECT FLOOR(balance/1000)`

Financial Domain Example

Let's say we have a **transactions** table:

transaction_id	account_id	transaction_type	amount	transaction_date
2001	1001	Deposit	500.00	2024-01-15
2002	1001	Withdrawal	-200.00	2024-01-16
2003	1002	Deposit	1000.00	2024-01-17

Query: Calculate total deposits, average transaction amount, and count of transactions per account

```sql
SELECT
    account_id,
    COUNT(*) as total_transactions,
    SUM(CASE WHEN amount > 0 THEN amount ELSE 0 END) as total_deposits,
    AVG(ABS(amount)) as avg_transaction_amount,
    ROUND(SUM(amount), 2) as net_amount
FROM transactions
GROUP BY account_id
HAVING COUNT(*) > 1
ORDER BY net_amount DESC;
```

Key Takeaways

1. **Clauses** structure your SQL query (SELECT, FROM, WHERE, etc.)

2. **Operators** help you create conditions and perform calculations
3. **Functions** transform and manipulate your data
4. Always use proper syntax and understand the order of execution
5. Practice with real financial data scenarios to master these concepts

Practice Tips

- Start with simple queries and gradually add complexity
- Use financial datasets to practice (bank accounts, transactions, loans)
- Always test your queries with sample data first
- Pay attention to data types when using functions and operators