**Beginner Level (Assignments 1–8)**

1. **Basic Selection and Filtering**
   * Write a query to display all customer details (SELECT, FROM, SELECT \*).
   * Filter customers to show only those who live in "California" (WHERE, <>).
2. **Sorting and Limiting Results**
   * Display the top 5 most expensive products (SELECT, TOP, ORDER BY).
   * Retrieve the top 10% of products by price (PERCENT).
3. **Using Aliases and DISTINCT**
   * Retrieve a list of unique product categories and alias the column as "Category Type" (DISTINCT, AS).
4. **Combining Criteria**
   * Retrieve all customers from either "Texas" or "Florida" (WHERE, IN).
   * Filter customers who spent more than $500 and purchased in 2023 (AND, BETWEEN).
5. **NULL Handling**
   * Find all products where the description is missing (IS NULL).
   * List all orders where comments are present (IS NOT NULL).
6. **Pattern Matching**
   * Find customers whose names start with "A" or end with "s" (LIKE, Wildcards).
7. **Basic Aggregations**
   * Calculate the total revenue from the order\_items table (SUM).
   * Find the average price of products and the highest price in each category (AVG, MAX, GROUP BY).
8. **String Functions**
   * Display the first 3 characters of product names (LEFT).
   * Find the position of the word "Bike" in product names (CHARINDEX).

**Intermediate Level (Assignments 9–16)**

1. **Substring and Replace**
   * Extract the last 4 digits of customers’ phone numbers (RIGHT, SUBSTRING).
   * Replace "Bike" with "Cycle" in all product names (REPLACE).
2. **Advanced Filtering**
   * List all orders placed between January 1, 2023, and March 31, 2023 (BETWEEN).
   * Exclude all customers who live in "Nevada" or "Alaska" (<>, NOT IN).
3. **Combining Data from Two Tables**
   * Retrieve customer names along with their order details (INNER JOIN).
   * Show products and their associated categories (LEFT JOIN).
4. **Aggregation with HAVING**
   * Display categories with total sales exceeding $10,000 (GROUP BY, HAVING).
   * Find stores with more than 50 orders placed (COUNT, HAVING).
5. **Advanced Sorting**
   * List products sorted by the length of their names (ORDER BY Length of Alphabets).
   * Sort orders based on the second column in the result (ORDER BY with Column Indexes).
6. **DML (Data Manipulation Language)**
   * Insert a new customer into the customers table.
   * Update a product's price by increasing it by 10%.
   * Delete all records from the stocks table where quantity is 0.
7. **Pagination**
   * Retrieve the 6th to 10th most expensive products (OFFSET, FETCH).
8. **Combining String Functions**
   * Find all customers whose phone numbers contain "123" and extract only the digits before it (CHARINDEX, SUBSTRING).

**Advanced Level (Assignments 17–25)**

1. **Subqueries**
   * Find all orders where the total amount exceeds the average order value (SUBQUERY, AVG).
   * List customers who haven’t placed any orders.
2. **Complex Joins**
   * Retrieve all products and their stock details, including stores with no stock for that product (LEFT JOIN).
3. **Key Concepts**
   * Use foreign keys to retrieve all orders for a specific customer.
4. **Advanced Filtering with Aggregations**
   * Retrieve the top 5 customers with the highest total purchase value (SUM, ORDER BY, TOP).
   * Find all categories where the average price of products is below $100.

Here are **20 intermediate SQL practice assignments** based on the database schema in the provided image:

**1. Retrieve customer details for a specific city**

List all customers who live in "Chicago".

**2. Find customers based on multiple states**

Retrieve all customers located in either "California" or "New York".

**3. Filter products by price range**

Retrieve all products whose price is between 100 and 500.

**4. Search staff email addresses**

Find all staff members whose email contains "sales".

**5. Use wildcards for product names**

List all products whose names start with "Pro" and end with any two characters.

**6. Group orders by status**

Group orders by order\_status and display the total number of orders in each status.

**7. Filter groups based on totals**

Group orders by staff\_id and include only those staff members who have processed more than 10 orders.

**8. Retrieve customers and their orders**

List all customers and their associated orders, including customers who have not placed any orders.

**9. Join products and their categories**

Retrieve product names along with their category names.

**10. Search for addresses with specific keywords**

Find all stores whose street address contains the word "Main".

**11. Extract parts of data**

Retrieve the first three characters of the zip\_code of all customers.

**12. Modify store addresses**

Replace "Street" with "St." in the address of all stores.

**13. Sort orders**

List all orders sorted by their shipped\_date in descending order.

**14. Pagination**

Retrieve the first 5 most expensive products, skipping the first 5.

**15. Analyze stock levels**

Find the total quantity of all products in stock for each store.

**16. Multi-table analysis**

List all products, their quantities in stock, and the store names where they are available.

**17. Filter phone numbers**

Retrieve all customers whose phone numbers start with "555".

**18. Category price analysis**

Calculate the average, minimum, and maximum product prices for each category.

**19. Orders within a time frame**

Retrieve all orders placed in the last 3 months.

**20. Discounted order items**

Retrieve all order items where the discount is greater than 0.10 and sort them by the highest discount.

Let me know if you'd like SQL code for any of these!