Accessing data

1. **Retrieve company details:**

```
SELECT company_name, contact_name, phone
FROM customers;
```

   - `SELECT`: Used to specify the columns to retrieve from a table.
   - `FROM`: Indicates the table from which to select the data.
   - Fetches `company_name`, `contact_name`, and `phone` from the `customers` table.

2. **Retrieve territory details:**

```
SELECT territory_id, territory_description
FROM territories;
```

   - Fetches `territory_id` and `territory_description` from the `territories` table.

3. **Retrieve specific territories:**

```
SELECT territory_id, territory_description
FROM territories
WHERE TerritoryDescription LIKE 'W%';
```

   - `WHERE`: Used to filter records based on specified conditions.
   - `LIKE`: Used in a `WHERE` clause to search for a specified pattern.
   - Fetches `territory_id` and `territory_description` for territories where the description starts with 'W'.

4. **Count territories by region:**

```
SELECT COUNT(territory_id)
FROM territories
WHERE region_id > 1
GROUP BY region_id;
```

- `COUNT`: An aggregate function that returns the number of rows that matches a specified condition.
- `GROUP BY`: Groups rows that have the same values into summary rows.
- Counts the number of territories for each `region_id` greater than 1.

5. **Count territories with a condition:**

```
SELECT COUNT(territory_id)
FROM territories
WHERE region_id > 1
GROUP BY region_id
HAVING COUNT(region_id) > 10;
```

- `HAVING`: Used to filter records that work on aggregated data.
- Counts the number of territories for each `region_id` greater than 1, only if the count is more than 10.

6. **Ordered count of territories:**

```
SELECT COUNT(territory_id)
FROM territories
WHERE region_id > 1
GROUP BY region_id
HAVING COUNT(territory_id) > 10
ORDER BY COUNT(territory_id);
```

- `ORDER BY`: Used to sort the result set in ascending or descending order.
- Similar to the previous query, but the results are ordered by the count of territories.

7. **Retrieve order details:**

```
SELECT order_id, customer_id, employee_id, order_date, freight
FROM orders;
```

- Fetches `order_id`, `customer_id`, `employee_id`, `order_date`, and `freight` from the `orders` table.

8. **Retrieve company names and addresses:**

```sql
SELECT company_name, address FROM customers;
```

- Fetches `company_name` and `address` from the `customers` table.

9. **Retrieve supplier contact details:**

```sql
SELECT company_name, contact_name FROM suppliers;
```

- Fetches `company_name` and `contact_name` from the `suppliers` table.

10. **Retrieve product details:**

```sql
SELECT product_name, unit_price FROM products;
```

- Fetches `product_name` and `unit_price` from the `products` table.

11. **Retrieve employee personal details:**

```sql
SELECT last_name, first_name, birth_date, hire_date FROM employees;
```

- Fetches `last_name`, `first_name`, `birth_date`, and `hire_date` from the `employees` table.

12. **Retrieve specific orders:**

```sql
SELECT order_id, customer_id, employee_id, order_date, freight
FROM orders
WHERE customer_id LIKE 'C%';
```

- Fetches orders where `customer_id` starts with 'C'.

13. **Retrieve employees hired in 1992:**

```sql
SELECT last_name, first_name, address, city
FROM employees
WHERE hire_date
BETWEEN '1992-01-01' AND '1992-12-31';
```

- `BETWEEN`: Used to filter the result set within a certain range.
- Fetches employees hired in the year 1992.

14. **Retrieve employees from specific cities:**

```
SELECT last_name, first_name, address, city
FROM employees
WHERE city IN ('London', 'Seattle');
```

- `IN`: Allows specifying multiple values in a `WHERE` clause.
- Fetches employees from London or Seattle.

15. **Retrieve employees with specific last names:**

```
SELECT last_name, first_name, hire_date
FROM employees
WHERE last_name LIKE 'D_v%';
```

- Fetches employees with last names starting with 'D' followed by 'v' and any character.

```
SELECT last_name, first_name, hire_date
FROM employees
WHERE last_name LIKE '%\_%';
```

- Fetches employees with last names containing an underscore (_), using backslash () as the escape character.

```
SELECT last_name, first_name, hire_date
FROM employees
WHERE last_name LIKE '%!_%' ESCAPE '!';
```

- Fetches employees with last names containing an underscore (_), using exclamation mark (!) as the escape character.

16. **Retrieve employee by ID:**

```
SELECT *
FROM employees
WHERE employee_id = 8;
```

- Fetches all details of the employee with `employee_id` 8.

17. **Retrieve employees from London:**

```
SELECT first_name, last_name
FROM employees
WHERE city = 'London';
```

- Fetches `first_name` and `last_name` of employees located in London.

18. **Retrieve employees born before 1969:**

```
SELECT first_name, last_name, birth_date
FROM employees
WHERE birth_date < '1969-01-01';
```

- Fetches `first_name`, `last_name`, and `birth_date` of employees born before January 1, 1969.

```
SELECT first_name, last_name, birth_date
FROM employees
WHERE birth_date::date < '1969-01-01';
```

- Similar to the previous query, but explicitly casts `birth_date` to a date type.

```
SELECT first_name, last_name, EXTRACT(year from AGE('2024-01-01',
birth_date)) AS age
FROM employees
WHERE EXTRACT(year from AGE('2024-01-01', birth_date)) > 55;
```

- **`EXTRACT`**: Extracts a part of a date or time.
- **`AGE`**: Calculates the age between two dates.
- Fetches `first_name`, `last_name`, and calculated age of employees who will be older than 55 years on January 1, 2024.

19. **Retrieve products within a price range:**

```
SELECT *
FROM products
```

```
WHERE unit_price
BETWEEN 10 AND 50;
```

- Fetches all details of products with `unit_price` between 10 and 50.

20. **Count and group orders:**

```
SELECT employee_id,
  EXTRACT(year from order_date) AS order_year,
  COUNT(customer_id) AS num_order
FROM orders
WHERE customer_id like 'C%'
GROUP BY employee_id, order_year
ORDER BY order_year;
```

- `SUM`: An aggregate function that returns the sum of a numeric column.
- Fetches order statistics by `employee_id` and order year, including counts and total freight, with some conditions and ordering.

21. **Count and sum freight for orders:**

```
SELECT employee_id,
  EXTRACT(year from order_date) AS order_year,
  COUNT(customer_id) AS num_order,
  SUM(freight) AS totalfreight
FROM orders
WHERE customer_id LIKE 'C%'
GROUP BY employee_id, order_year
ORDER BY order_year;
```

- Fetches order statistics by `employee_id` and order year, including counts and total freight, with some conditions and ordering.

22. **Count and sum freight for orders with having clause:**

```
SELECT employee_id,
  EXTRACT(year from order_date) AS order_year,
  COUNT(customer_id) AS num_order,
  SUM(freight) AS totalfreight
FROM orders
WHERE customer_id LIKE 'C%'
GROUP BY employee_id, order_year
```

```
HAVING COUNT(customer_id) > 1
ORDER BY order_year;
```

- Fetches order statistics by `employee_id` and order year, including counts and total freight, with some conditions and ordering.

23. **Retrieve products with specific criteria:**

```
SELECT *
FROM products
WHERE product_name LIKE 'N%'
AND unit_price > 50;
```

- Fetches all details of products with `product_name` starting with 'N' and `unit_price` greater than 50.

24. **Count employees by city:**

```
SELECT city, COUNT(*)
FROM employees
GROUP BY city;
```

- Fetches the count of employees grouped by `city`.

25. **Retrieve suppliers from specific city:**

```
SELECT *
FROM suppliers
WHERE company_name LIKE 'A%'
AND city = 'London';
```

- Fetches all details of suppliers with `company_name` starting with 'A' and located in London.

26. **Count customers with specific criteria:**

```
SELECT COUNT(*)
FROM customers
WHERE city LIKE 'México%' AND contact_title = 'Owner';
```

- Counts the number of customers in cities starting with 'México' and having the contact title 'Owner'.

27. **Retrieve distinct employee orders by year:**

```
SELECT DISTINCT employee_id,
  extract(year from order_date) as order_year
FROM orders
WHERE customer_id LIKE 'C%';
```

- `DISTINCT`: Removes duplicate rows from the result set.
- Fetches distinct `employee_id` and order year for orders where `customer_id` starts with 'C'.

28. **Retrieve distinct employee orders with count:**

```
SELECT DISTINCT employee_id,
  extract(year from order_date) as order_year,
  COUNT(*) AS numorders
FROM orders
WHERE customer_id LIKE 'C%'
GROUP BY employee_id, order_year
HAVING COUNT(*) > 1
ORDER BY employee_id, order_year DESC;
```

- Fetches distinct `employee_id` and order year with the count of orders greater than 1, ordered by `employee_id` and order year in descending order.