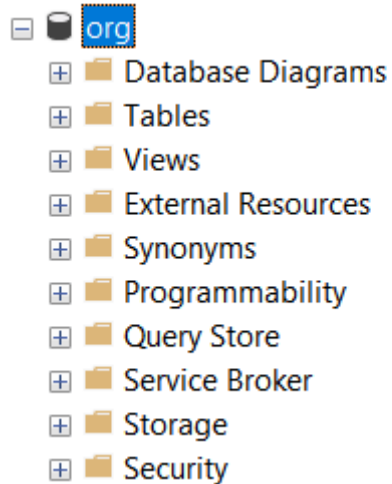


SQL QUERIES / JOINS / AGGREGATE FUNCTIONS

1. Database Creation and Setup

- Drop and Create Database: Ensure a fresh database by dropping if it exists, then creating it.


```
DROP DATABASE IF EXISTS org;  
CREATE DATABASE org;  
USE org;
```



2. Creating the **employees** Table

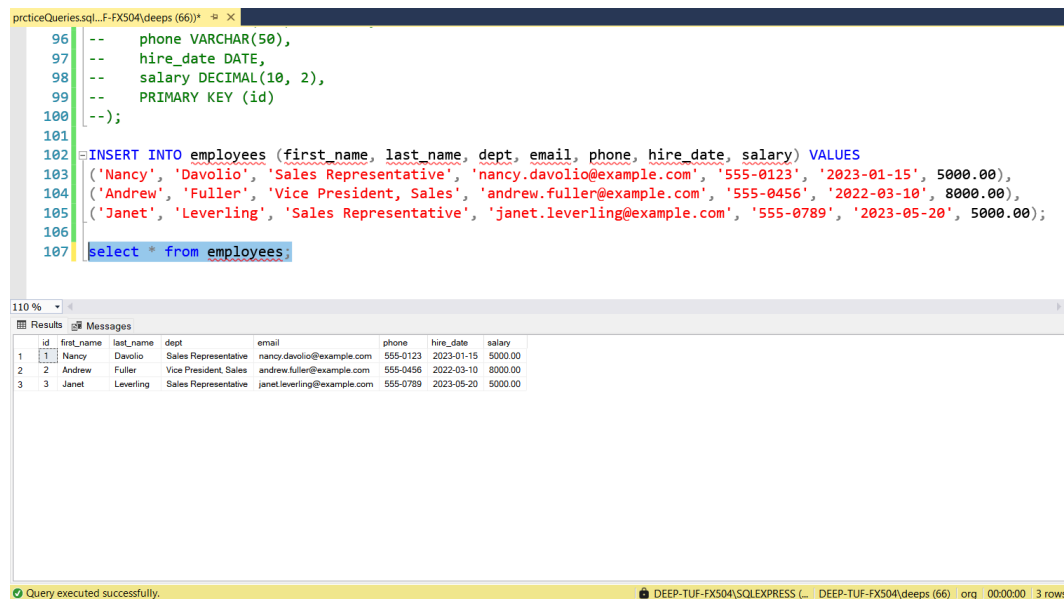
- Define Fields and Primary Key: Set up columns and constraints in **employees**.

```
CREATE TABLE employees (  
  id INT NOT NULL IDENTITY(1,1),  
  first_name VARCHAR(255) NOT NULL,  
  last_name VARCHAR(255),  
  dept VARCHAR(255),  
  email VARCHAR(255) NOT NULL,  
  phone VARCHAR(50),  
  hire_date DATE,  
  salary DECIMAL(10, 2),  
  PRIMARY KEY (id)  
);
```

 **dbo.employees**

3. Inserting Employee Records

- **Add Sample Data:** Insert multiple records into `employees`.



```
prcticeQueries.sql...F-FX504\deeps (66)*  
96 -- phone VARCHAR(50),  
97 -- hire_date DATE,  
98 -- salary DECIMAL(10, 2),  
99 -- PRIMARY KEY (id)  
100 --);  
101  
102 INSERT INTO employees (first_name, last_name, dept, email, phone, hire_date, salary) VALUES  
103 ('Nancy', 'Davolio', 'Sales Representative', 'nancy.davolio@example.com', '555-0123', '2023-01-15', 5000.00),  
104 ('Andrew', 'Fuller', 'Vice President, Sales', 'andrew.fuller@example.com', '555-0456', '2022-03-10', 8000.00),  
105 ('Janet', 'Leverling', 'Sales Representative', 'janet.leverling@example.com', '555-0789', '2023-05-20', 5000.00);  
106  
107 select * from employees;
```

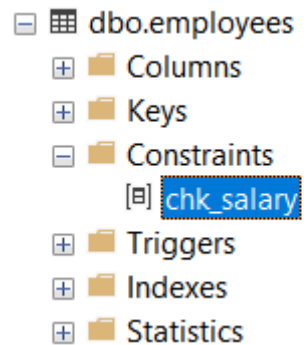
	id	first_name	last_name	dept	email	phone	hire_date	salary
1	1	Nancy	Davolio	Sales Representative	nancy.davolio@example.com	555-0123	2023-01-15	5000.00
2	2	Andrew	Fuller	Vice President, Sales	andrew.fuller@example.com	555-0456	2022-03-10	8000.00
3	3	Janet	Leverling	Sales Representative	janet.leverling@example.com	555-0789	2023-05-20	5000.00

Query executed successfully. DEEP-TUF-FX504\SQLEXPRESS [DEEP-TUF-FX504\deeps (66)] org 00:00:00 3 rows

4. Data Integrity Constraints

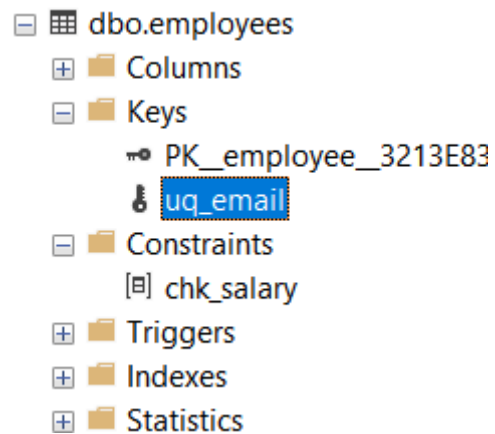
- **Enforce Minimum Salary:** Add a constraint to prevent salaries below 3000.

```
ALTER TABLE employees ADD CONSTRAINT chk_salary CHECK (salary >= 3000);
```



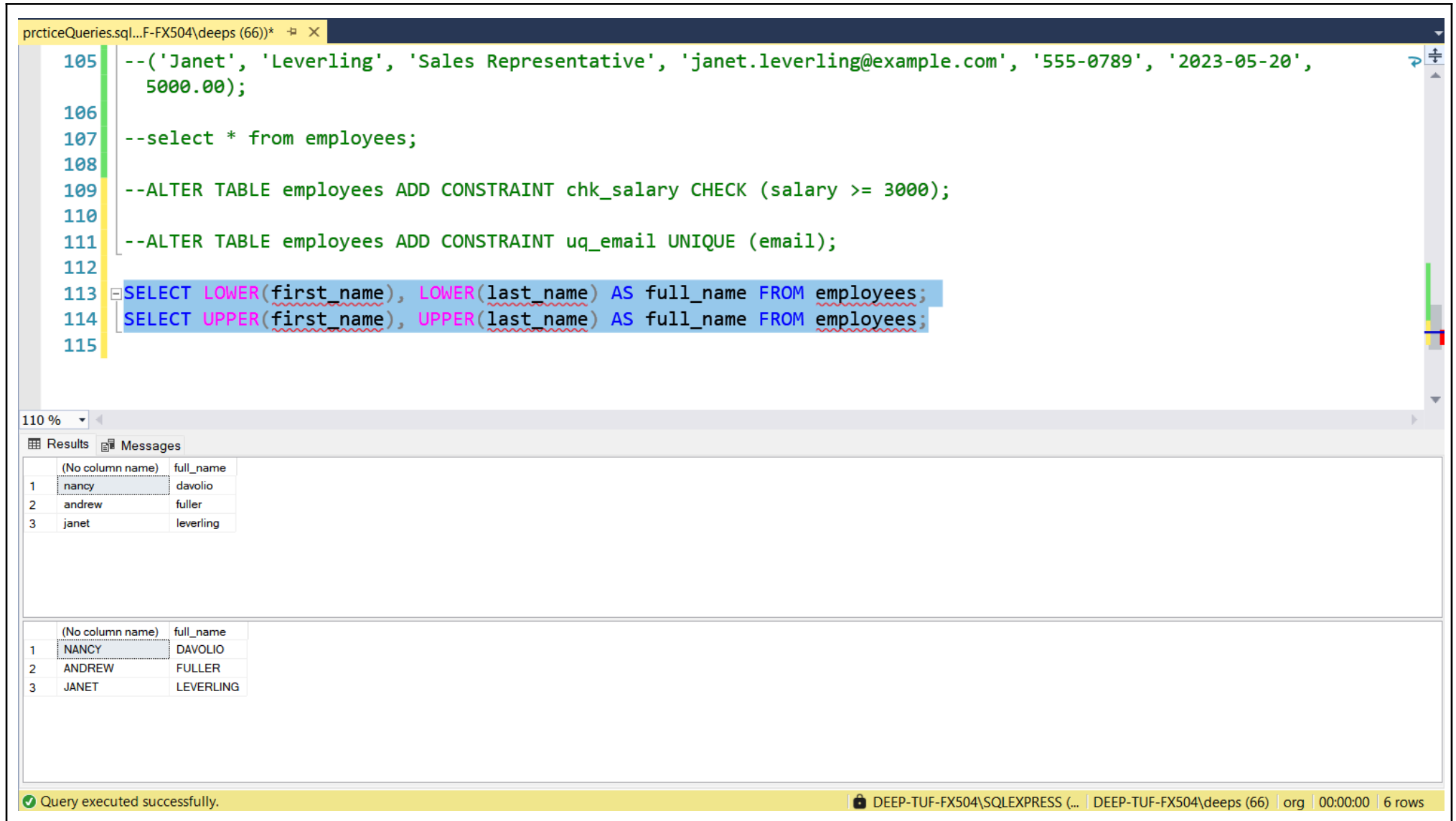
- **Email Uniqueness:** Add a unique constraint to the `email` column.

```
ALTER TABLE employees ADD CONSTRAINT uq_email UNIQUE (email);
```



5. String Manipulation Functions

- Convert Names to Lower and Upper Case:



The screenshot shows a SQL IDE window with a query editor and a results pane. The query editor contains the following SQL code:

```
105 --('Janet', 'Leverling', 'Sales Representative', 'janet.leverling@example.com', '555-0789', '2023-05-20',  
106 5000.00);  
107 --select * from employees;  
108  
109 --ALTER TABLE employees ADD CONSTRAINT chk_salary CHECK (salary >= 3000);  
110  
111 --ALTER TABLE employees ADD CONSTRAINT uq_email UNIQUE (email);  
112  
113 SELECT LOWER(first_name), LOWER(last_name) AS full_name FROM employees;  
114 SELECT UPPER(first_name), UPPER(last_name) AS full_name FROM employees;  
115
```

The results pane shows two tables. The first table is the result of the first query (line 113), and the second table is the result of the second query (line 114).

	(No column name)	full_name
1	nancy	davolio
2	andrew	fuller
3	janet	leverling

	(No column name)	full_name
1	NANCY	DAVOLIO
2	ANDREW	FULLER
3	JANET	LEVERLING

The status bar at the bottom indicates: Query executed successfully. DEEP-TUF-FX504\SQLEXPRESS (...) DEEP-TUF-FX504\deeps (66) org 00:00:00 6 rows

- Concatenate First and Last Names:

```

114 | --SELECT UPPER(first_name), UPPER(last_name) AS full_name FROM employees;
115 |
116 | SELECT CONCAT(first_name, ' ', last_name) AS full_name FROM employees;
117 |

```

110 %

Results Messages

	full_name
1	Nancy Davolio
2	Andrew Fuller
3	Janet Leverling

- Extract Email Domain:

```

118 | SELECT first_name, last_name, email, SUBSTRING(email, CHARINDEX('@', email) + 1, LEN(email)) AS domain
119 | FROM employees;
120 |

```

110 %

Results Messages

	first_name	last_name	email	domain
1	Nancy	Davolio	nancy.davolio@example.com	example.com
2	Andrew	Fuller	andrew.fuller@example.com	example.com
3	Janet	Leverling	janet.leverling@example.com	example.com

6. Date Functions and Tenure Calculations

- Extract Year, Month, Day from Hire Date:

```
121 SELECT first_name, last_name, hire_date,
122        YEAR(hire_date) AS hire_year,
123        MONTH(hire_date) AS hire_month,
124        DAY(hire_date) AS hire_day
125 FROM employees;
126
127
```

110 %

Results Messages

	first_name	last_name	hire_date	hire_year	hire_month	hire_day
1	Nancy	Davolio	2023-01-15	2023	1	15
2	Andrew	Fuller	2022-03-10	2022	3	10
3	Janet	Leverling	2023-05-20	2023	5	20

- Calculate Tenure in Days:

```
126
127 SELECT first_name, last_name, DATEDIFF(DAY, hire_date, GETDATE()) AS tenure_days FROM employees;
128
```

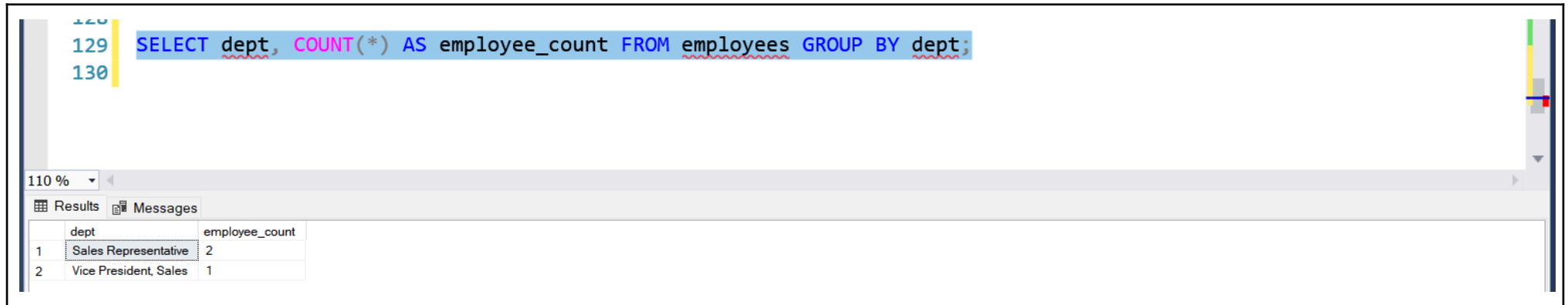
110 %

Results Messages

	first_name	last_name	tenure_days
1	Nancy	Davolio	654
2	Andrew	Fuller	965
3	Janet	Leverling	529

7. Aggregation and Summaries

- Count Employees by Department:



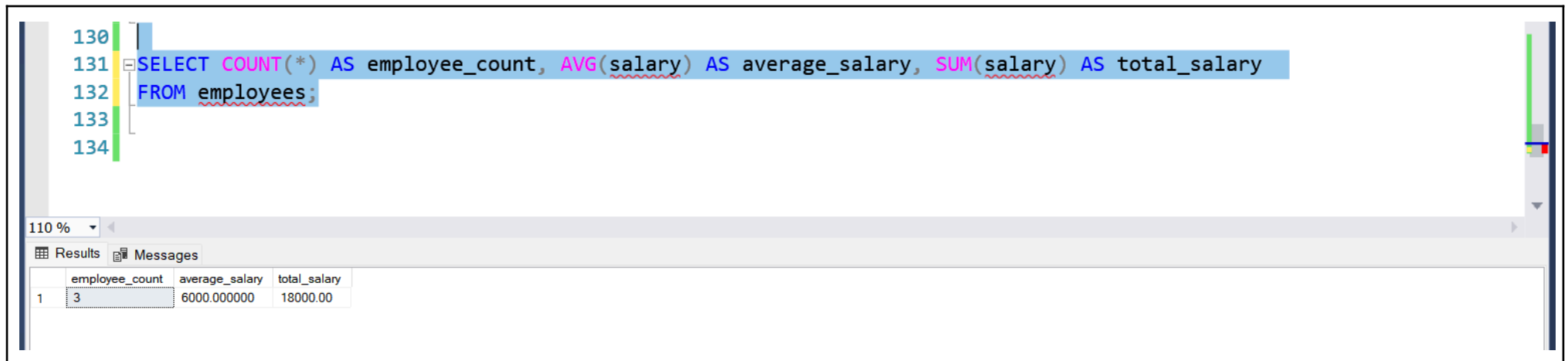
The screenshot shows a SQL query editor with the following code:

```
129 SELECT dept, COUNT(*) AS employee_count FROM employees GROUP BY dept;
```

Below the editor, the 'Results' tab is active, displaying a table with the following data:

	dept	employee_count
1	Sales Representative	2
2	Vice President, Sales	1

- Calculate Average and Total Salaries:



The screenshot shows a SQL query editor with the following code:

```
131 SELECT COUNT(*) AS employee_count, AVG(salary) AS average_salary, SUM(salary) AS total_salary  
132 FROM employees;
```

Below the editor, the 'Results' tab is active, displaying a table with the following data:

	employee_count	average_salary	total_salary
1	3	6000.000000	18000.00

8. Creating the **customers** Table with Foreign Key

- Define Customer Table with Foreign Key: Links customers to employees using `employee_id`.

```

CREATE TABLE customers (
  id INT NOT NULL IDENTITY(1,1) PRIMARY KEY,
  employee_id INT,
  company_name VARCHAR(255) NOT NULL,
  address VARCHAR(255),
  city VARCHAR(255),
  region VARCHAR(255),
  country VARCHAR(255),
  phone VARCHAR(255),
  FOREIGN KEY (employee_id) REFERENCES employees(id) ON DELETE SET NULL
);

```

```

INSERT INTO customers (employee_id, company_name, address, city, region, country, phone) VALUES
(1, 'Rancho Grande', 'Av. del Libertador 900', 'Buenos Aires', NULL, 'Argentina', '(1) 123-5555'),
(2, 'Alfreds Futterkiste', 'Obere Str. 57', 'Berlin', NULL, 'Germany', '030-0074321'),
(3, 'Bottom-Dollar Markets', '23 Tsawassen Blvd.', 'Tsawassen', 'BC', 'Canada', '(604) 555-4729'),
(1, 'Familia Arquibaldo', 'Rua Orós, 92', 'São Paulo', 'SP', 'Brazil', '(11) 555-9857'),
(2, 'Queen Cozinha', 'Alameda dos Canários, 891', 'São Paulo', 'SP', 'Brazil', '(11) 555-1189'),
(3, 'Laughing Bacchus Wine Cellars', '1900 Oak St.', 'Vancouver', 'BC', 'Canada', '(604) 555-3392'),
(1, 'Lazy K Kountry Store', '12 Orchestra Terrace', 'Walla Walla', 'WA', 'USA', '(509) 555-7969'),
(2, 'Drachenblut Delikatessen', 'Walserweg 21', 'Aachen', NULL, 'Germany', '0241-039123'),
(3, 'Frankenversand', 'Berliner Platz 43', 'München', NULL, 'Germany', '089-0877310'),
(1, 'Du monde entier', '67, rue des Cinquante Otages', 'Nantes', NULL, 'France', '40.67.88.88');

```

110 %

Results Messages

	id	employee_id	company_name	address	city	region	country	phone
1	5	1	Rancho Grande	Av. del Libertador 900	Buenos Aires	NULL	Argentina	(1) 123-5555
2	6	2	Alfreds Futterkiste	Obere Str. 57	Berlin	NULL	Germany	030-0074321
3	7	3	Bottom-Dollar Markets	23 Tsawassen Blvd.	Tsawassen	BC	Canada	(604) 555-4729
4	8	1	Familia Arquibaldo	Rua Orós, 92	São Paulo	SP	Brazil	(11) 555-9857
5	9	2	Queen Cozinha	Alameda dos Canários, 891	São Paulo	SP	Brazil	(11) 555-1189
6	10	3	Laughing Bacchus Wine Cellars	1900 Oak St.	Vancouver	BC	Canada	(604) 555-3392
7	11	1	Lazy K Kountry Store	12 Orchestra Terrace	Walla Walla	WA	USA	(509) 555-7969
8	12	2	Drachenblut Delikatessen	Walserweg 21	Aachen	NULL	Germany	0241-039123
9	13	3	Frankenversand	Berliner Platz 43	München	NULL	Germany	089-0877310
10	14	1	Du monde entier	67, rue des Cinquante Otages	Nantes	NULL	France	40.67.88.88

9. Joining Tables

- **Inner Join:** Show employees with their customers.

```
159
160 SELECT e.first_name, e.last_name, c.company_name
161 FROM employees e
162 JOIN customers c ON e.id = c.employee_id;
163
```

110 %

Results Messages

	first_name	last_name	company_name
1	Nancy	Davolio	Rancho Grande
2	Andrew	Fuller	Alfreds Futterkiste
3	Janet	Leverling	Bottom-Dollar Markets
4	Nancy	Davolio	Familia Arquibaldo
5	Andrew	Fuller	Queen Cozinha
6	Janet	Leverling	Laughing Bacchus Wine Cellars
7	Nancy	Davolio	Lazy K Kountry Store
8	Andrew	Fuller	Drachenblut Delikatessen
9	Janet	Leverling	Frankenversand
10	Nancy	Davolio	Du monde entier

- **Left Join:** Show all employees, even those without customers.

```
164 SELECT e.id, e.first_name, e.last_name, c.company_name
165 FROM employees e
166 LEFT JOIN customers c ON e.id = c.employee_id;
167
```

110 %

Results Messages

	id	first_name	last_name	company_name
1	1	Nancy	Davolio	Rancho Grande
2	1	Nancy	Davolio	Familia Arquibaldo
3	1	Nancy	Davolio	Lazy K Kountry Store
4	1	Nancy	Davolio	Du monde entier
5	2	Andrew	Fuller	Alfreds Futterkiste
6	2	Andrew	Fuller	Queen Cozinha
7	2	Andrew	Fuller	Drachenblut Delikatessen
8	3	Janet	Leverling	Bottom-Dollar Markets
9	3	Janet	Leverling	Laughing Bacchus Wine Cellars
10	3	Janet	Leverling	Frankenversand

10. Customer Count per Employee

- Count Customers Each Employee Manages:

```
168 SELECT e.id, e.first_name, e.last_name, COUNT(c.id) AS customer_count
169 FROM employees e
170 LEFT JOIN customers c ON e.id = c.employee_id
171 GROUP BY e.id, e.first_name, e.last_name;
172
```

110 %

Results Messages

	id	first_name	last_name	customer_count
1	1	Nancy	Davolio	4
2	2	Andrew	Fuller	3
3	3	Janet	Leverling	3