SQL Assessment – Step-by-Step Guide with Explanations

Step 1: Create the Database

```
CREATE DATABASE CompanyDB;
```

This command initializes a new database named **CompanyDB**, which serves as a container for all tables and objects related to our assessment.

Step 2: Select the Database

```
USE CompanyDB;
```

Switches the context to **CompanyDB** so that all subsequent commands apply to this specific database rather than the server's default.

Step 3: Create the Tables

3.1 Customer Table

```
CREATE TABLE Customer (
  customer_id   INT,
  cust_name     VARCHAR(50),
  city          VARCHAR(50),
  grade         INT,
  salesman_id   INT
);
```

Defines the **Customer** table schema:

- customer_id uniquely identifies each customer.
- cust_name holds the customer's name.
- city records their location.
- grade represents a numeric rating.
- salesman_id links to the Salesman table

3.2 Salesman Table

```
CREATE TABLE Salesman (
  salesman_id   INT,
  name          VARCHAR(50),
  city          VARCHAR(50),
  commission    DECIMAL(4,2)
);
```

Defines the Salesman table schema:
- salesman_id uniquely identifies each salesperson.
- name stores the salesperson's name.
- city indicates their base of operations.
- commission records their commission rate as a decimal.

Step 4: Insert Sample Data

4.1 Customer Data

```
INSERT INTO Customer VALUES
  (3002, 'Nick Rimando',   'New York',   100, 5001),
  (3007, 'Brad Davis',     'New York',   200, 5001),
  (3005, 'Graham Zusi',    'California', 200, 5002),
  (3008, 'Julian Green',   'London',     300, 5002),
  (3004, 'Fabian Johnson', 'Paris',      300, 5006),
  (3009, 'Geoff Cameron',  'Berlin',     100, 5003),
  (3003, 'Jozy Altidor',   'Moscow',     200, 5007),
  (3001, 'Brad Guzan',     'London',     NULL,5005);
```

Populates **Customer** with eight rows, each representing a unique customer and their assigned salesman_id. A NULL grade demonstrates handling of missing values.

4.2 Salesman Data

```
INSERT INTO Salesman VALUES
  (5001, 'James Hoog',  'New York', 0.15),
  (5002, 'Nail Knite',  'Paris',    0.13),
  (5005, 'Pit Alex',    'London',   0.11),
  (5006, 'Mc Lyon',     'Paris',    0.14),
  (5007, 'Paul Adam',   'Rome',     0.13),
  (5003, 'Lauson Hen',  'San Jose', 0.12);
```

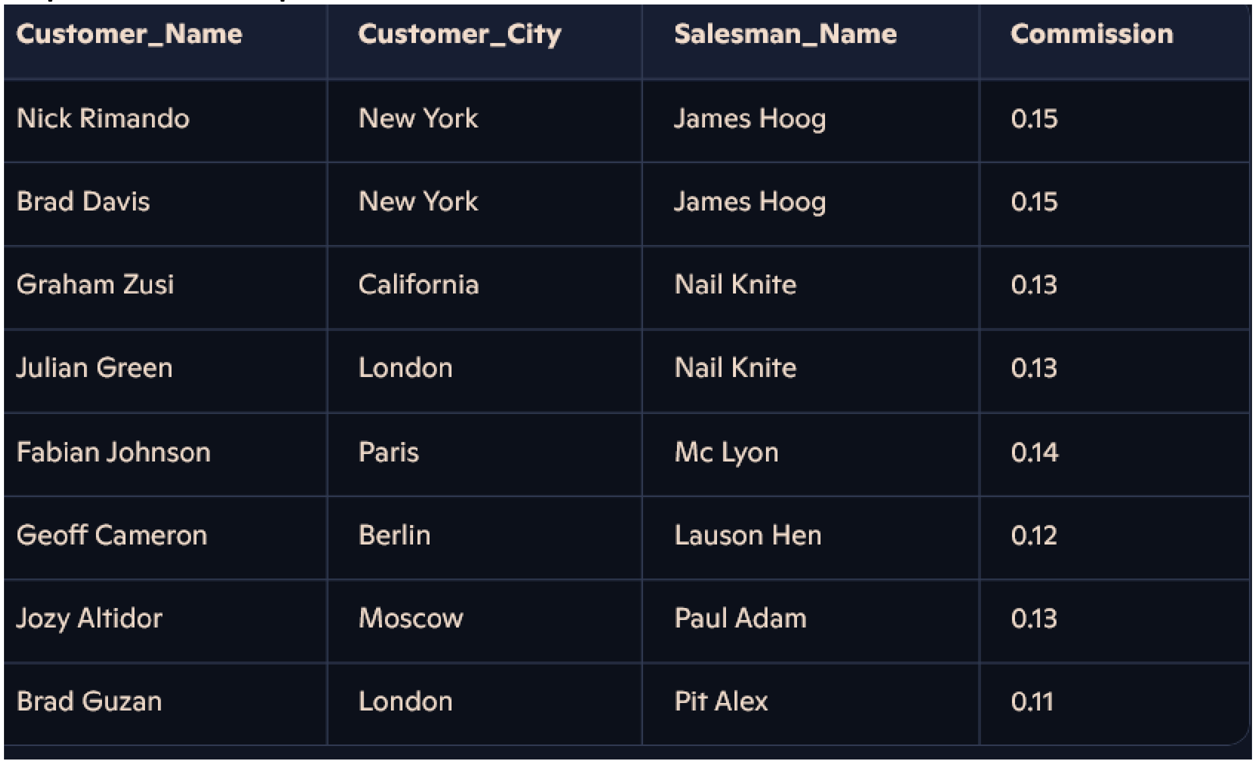
Populates **Salesman** with six rows, assigning each salesman_id a name, city, and commission rate.

Step 5: Write the JOIN Query

```
SELECT
  c.cust_name   AS Customer_Name,
  c.city        AS Customer_City,
  s.name        AS Salesman_Name,
  s.commission  AS Commission
FROM
  Customer c
INNER JOIN
  Salesman s
ON
  c.salesman_id = s.salesman_id;
```

This **INNER JOIN** matches records from **Customer** (c) to **Salesman** (s) where their salesman_id values align. Selected columns are aliased for readability.

Step 6: View the Output



Customer_Name	Customer_City	Salesman_Name	Commission
Nick Rimando	New York	James Hoog	0.15
Brad Davis	New York	James Hoog	0.15
Graham Zusi	California	Nail Knite	0.13
Julian Green	London	Nail Knite	0.13
Fabian Johnson	Paris	Mc Lyon	0.14
Geoff Cameron	Berlin	Lauson Hen	0.12
Jozy Altidor	Moscow	Paul Adam	0.13
Brad Guzan	London	Pit Alex	0.11

This result set lists each customer alongside their corresponding salesman and that salesman's commission rate, demonstrating the power of joining related tables.