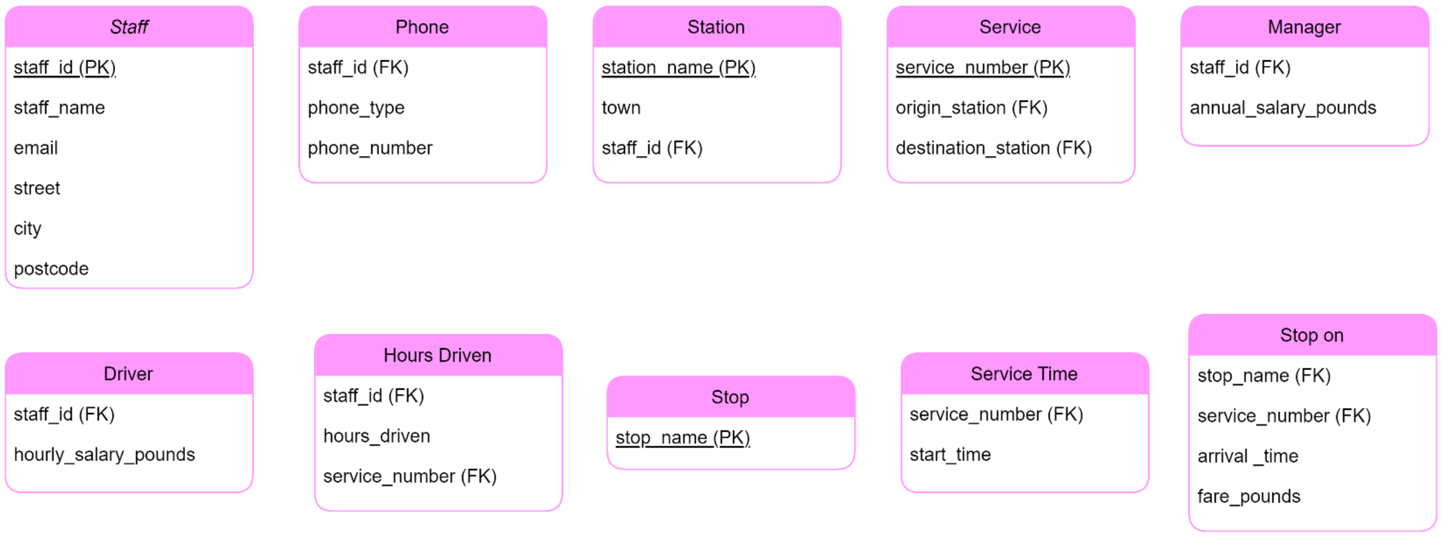
IS5102 - Database Management Systems

Practical 2: Database Management with SQLite

230012908

Task 1: Translation

The collection of relation schemas:



1. *Staff table*

   - 'staff_id`' is chosen as the primary key because it is unique for each staff member.

   - Other attributes 'staff_name', 'email', 'street', 'city', and 'postcode' are associated with staff members.

2. *Phone Table*

   - The 'staff_id' is a foreign key referencing the 'staff' table and establishing a relationship between staff and their phone numbers.

   – 'phone_type' is included to differentiate between different types of phone numbers (e.g., mobile, home).

   – 'phone_number' is the actual phone number, and it is marked as NOT NULL since it is an important piece of information.

3. *Station table*

   - 'station_name' is chosen as the primary key because each station should have a unique name.

   - 'staff_id' is a foreign key referencing the staff responsible for the station.

4. *Service table*

   - 'service_number' is the primary key that uniquely identifies each service.

   - 'origin_station' and 'destination_station' are foreign keys referencing the 'station' table, creating relationships between services and stations.

5. *Manager table*

   - 'staff_id' is a foreign key referencing the staff who is a manager.

   - 'annual_salary_pounds' represents the annual salary of the manager.

6. *Driver table*

   – 'staff_id' is a foreign key referencing the staff who is a driver.

   – 'hourly_salary_pounds' represents the hourly salary of the driver.

7. *Hours Driven table*

   - 'staff_id' and 'service_number' are foreign keys establishing relationships with the 'staff' and 'service' tables.

   - 'hours_driven' represents the number of hours a driver has driven a particular service.

8. *Stop table*

   - 'stop_name' is chosen as the primary key because each stop should have a unique name.

9. *Service Time table*

   - 'service_number' is a foreign key referencing the service to which the start time belongs.

   - 'start_time' represents the daily start time for a particular service.

10. *Stop on table*

    - 'stop_name' and 'service_number' are foreign keys establishing relationships with the 'stop' and 'service' tables.

- 'arrival_time' represents the arrival time at a particular stop for a specific service.

- 'fare_pounds' represents the fare charged for the service from the origin station to the stop.

Task 2: SQL Data Definition

Data definition was performed in DBeaver.

Database Design:

The first step was to design the database schema. This involved identifying entities, their attributes, and the relationships between them. The following tables were created: ‘staff’, ‘phone’, ‘station’, ‘service’, ‘manager’, ‘driver’, ‘hours_driven’, ‘stop’, ‘service_time’, and ‘stop_on’.

Establishing Relationships:

Foreign keys were used to establish relationships between tables. For example, in the ‘phone’ table, ‘staff_id’ is a foreign key referencing the ‘staff’ table. This ensures that phone numbers are associated with specific staff members.

```
CREATE TABLE phone (
     staff_id CHAR(5),
     phone_type CHAR(10),
     phone_number CHAR(10) NOT NULL,
     FOREIGN KEY (staff_id) REFERENCES staff
);
```

Ensuring Data Integrity:

Primary keys are carefully chosen to ensure uniqueness within a table. For instance, in the ‘staff’ table, ‘staff_id’ is the primary key. This guarantees that each staff member is uniquely identified.

```
CREATE TABLE staff (
     staff_id CHAR(5),
     staff_name VARCHAR(20) NOT NULL,
     email CHAR(13),
     street VARCHAR(30),
     city VARCHAR(30),
     postcode CHAR(6),
     PRIMARY KEY (staff_id)
);
```

Data Validation:

Data types and constraints are utilised to validate and restrict the input data. For instance, the ‘hours_driven’ table has a constraint to ensure that the number of hours driven is within a reasonable range.

```
CREATE TABLE hours_driven (
	staff_id CHAR(5),
	hours_driven NUMERIC(2),
	   service_number CHAR(3),
	FOREIGN KEY (staff_id) REFERENCES staff,
	   FOREIGN KEY (service_number) REFERENCES service
);
```

Testing:

The SQL code is thoroughly tested to ensure that it creates the tables and relationships correctly. This involves making queries to verify that the results align with expectations.

```
SELECT arrival_time AS 'Arrival Time', origin_station AS 'Origin
Station',  destination_station AS 'Destination Station',
service_number AS 'Service Number'
FROM stop_on NATURAL JOIN service
WHERE stop_name = "Buchanan Gardens, St Andrews" AND
arrival_time BETWEEN '10:00' AND '14:00'
ORDER BY arrival_time;
```

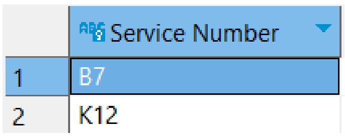
Documentation:

The SQL code and database schema are documented for clarity and reference. This includes comments within the code to explain the purpose of each chunk of code.

Task 3: SQL Data Manipulation

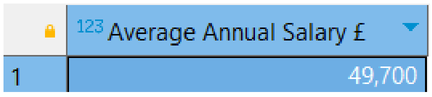
Data manipulation was performed in DBeaver.

```
-- TASK 3: SQL DATA MANIPULATION
-- QUERIES

-- 1 Listing all services which have Seagate Bus Station in Dundee as their
origin.

SELECT service_number AS 'Service Number'
FROM service
WHERE origin_station = 'Seagate Bus Station';
```



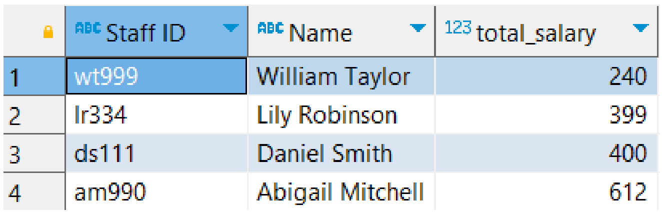
	Service Number
1	B7
2	K12

```sql
-- 2 Calculating an average monthly salary of a bus station manager.

SELECT AVG(annual_salary_pounds) AS 'Average Annual Salary £'
FROM manager;
```



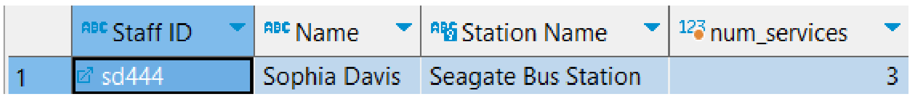
	Average Annual Salary £
1	49,700

```sql
-- 3 Listing the names of all drivers of services which have Edinburgh Bus
Station in Edinburgh as their origin or destination, in increasing order of
the amount to be paid to them for the hours driven.

SELECT DISTINCT driver.staff_id AS 'Staff ID', staff.staff_name AS 'Name',
SUM(driver.'hourly_salary_pounds' * hours_driven.hours_driven) AS
total_salary
FROM driver JOIN hours_driven ON driver.staff_id = hours_driven.staff_id
JOIN service ON hours_driven.service_number = service.service_number
JOIN staff ON driver.staff_id = staff.staff_id
JOIN station ON service.destination_station = station.station_name OR
service.origin_station = station.station_name
WHERE station.town = 'Edinburgh'
GROUP BY driver.staff_id, staff.staff_name
ORDER BY total_salary;
```



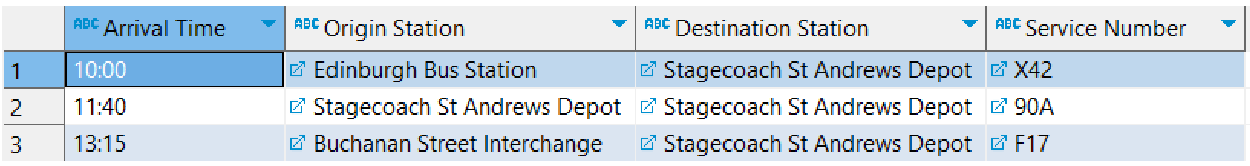
	Staff ID	Name	total_salary
1	wt999	William Taylor	240
2	lr334	Lily Robinson	399
3	ds111	Daniel Smith	400
4	am990	Abigail Mitchell	612

```sql
-- 4 Listing the manager of the most connected station, measured by the
number of services which have that station as their origin or destination.

SELECT manager.staff_id AS 'Staff ID', staff.staff_name AS 'Name',
station.station_name AS 'Station Name', COUNT(*) AS num_services
FROM station
JOIN service ON station.station_name = service.destination_station OR
station.station_name = service.origin_station
JOIN manager ON station.staff_id = manager.staff_id
JOIN staff ON manager.staff_id = staff.staff_id
GROUP BY manager.staff_id = staff.staff_name, station.station_name
ORDER BY num_services DESC
LIMIT 1;
```



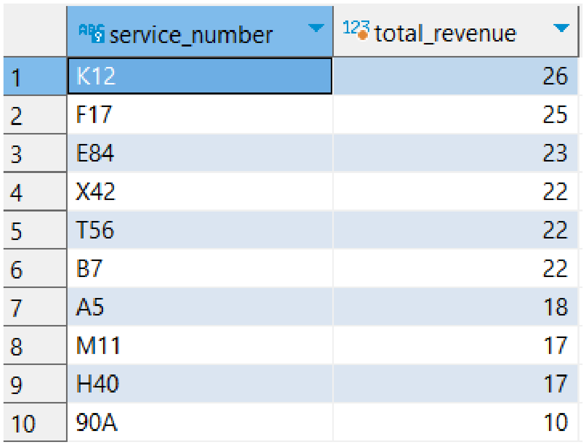
	Staff ID	Name	Station Name	num_services
1	sd444	Sophia Davis	Seagate Bus Station	3

```sql
-- 5 For the bus stop "Buchanan Gardens, St Andrews" listing in the
chronological order arrival times at this stop, origins, destinations, and
service numbers of all bus services passing this stop between 10 am and 2 pm.

SELECT arrival_time AS 'Arrival Time', origin_station AS 'Origin Station',
destination_station AS 'Destination Station', service_number AS 'Service
Number'
FROM stop_on NATURAL JOIN service
WHERE stop_name = "Buchanan Gardens, St Andrews" AND arrival_time BETWEEN
'10:00' AND '14:00'
ORDER BY arrival_time;
```



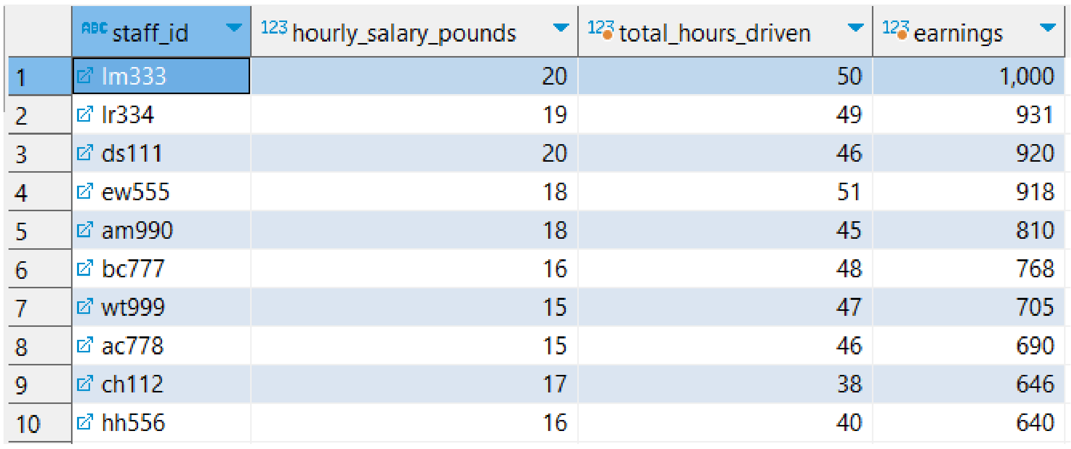
	Arrival Time	Origin Station	Destination Station	Service Number
1	10:00	Edinburgh Bus Station	Stagecoach St Andrews Depot	X42
2	11:40	Stagecoach St Andrews Depot	Stagecoach St Andrews Depot	90A
3	13:15	Buchanan Street Interchange	Stagecoach St Andrews Depot	F17

```
-- 6 This query calculates the total revenue generated by each bus service
based on the sum of fares collected from each stop.

SELECT
    s.service_number,
    SUM(so.fare_pounds) AS total_revenue
FROM service s
JOIN stop_on so ON s.service_number = so.service_number
GROUP BY s.service_number
ORDER BY total_revenue DESC;
```



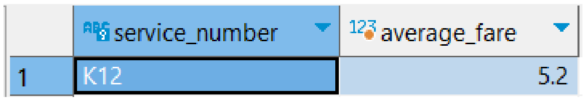
	service_number	total_revenue
1	K12	26
2	F17	25
3	E84	23
4	X42	22
5	T56	22
6	B7	22
7	A5	18
8	M11	17
9	H40	17
10	90A	10

```
-- 7 This query calculates the earnings for each driver based on their hourly
salary and the total hours driven.

SELECT
    d.staff_id,
    d.hourly_salary_pounds,
    SUM(hd.hours_driven) AS total_hours_driven,
    SUM(hd.hours_driven) * d.hourly_salary_pounds AS earnings
FROM driver d
JOIN hours_driven hd ON d.staff_id = hd.staff_id
GROUP BY d.staff_id
ORDER BY earnings DESC;
```



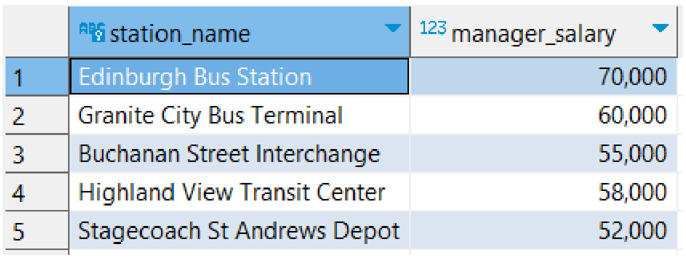
	staff_id	hourly_salary_pounds	total_hours_driven	earnings
1	lm333	20	50	1,000
2	lr334	19	49	931
3	ds111	20	46	920
4	ew555	18	51	918
5	am990	18	45	810
6	bc777	16	48	768
7	wt999	15	47	705
8	ac778	15	46	690
9	ch112	17	38	646
10	hh556	16	40	640

```
-- 8 This query identifies the bus service with the highest average fare.

SELECT
    s.service_number,
    AVG(so.fare_pounds) AS average_fare
FROM service s
JOIN stop_on so ON s.service_number = so.service_number
GROUP BY s.service_number
ORDER BY average_fare DESC
LIMIT 1;
```



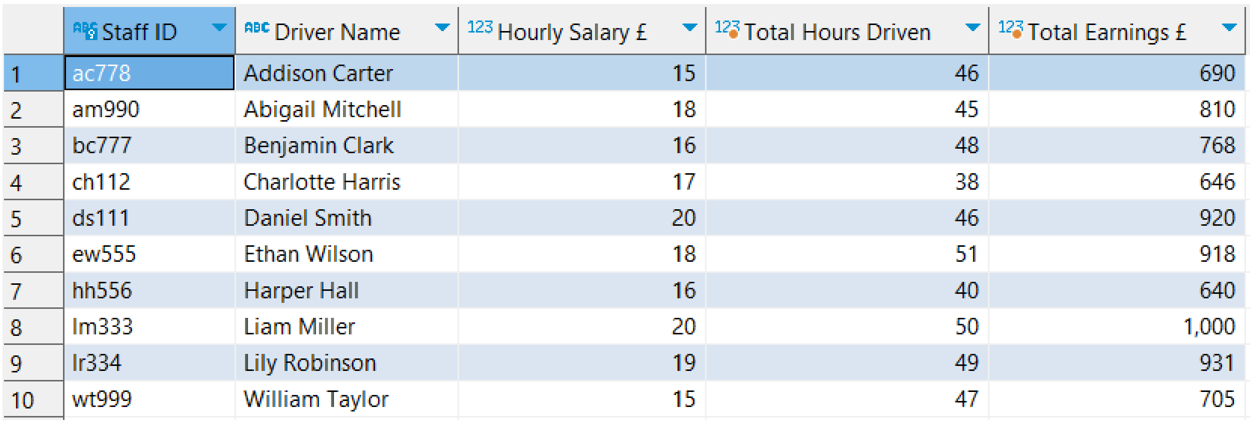
	service_number	average_fare
1	K12	5.2

```
-- 9 This query lists stations where the managers earn more than the average
salary.

SELECT
    st.station_name,
    m.annual_salary_pounds AS manager_salary
FROM station st
JOIN manager m ON st.staff_id = m.staff_id
WHERE m.annual_salary_pounds > (SELECT AVG(annual_salary_pounds) FROM
manager);
```



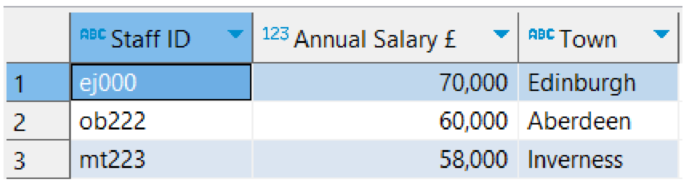
	station_name	manager_salary
1	Edinburgh Bus Station	70,000
2	Granite City Bus Terminal	60,000
3	Buchanan Street Interchange	55,000
4	Highland View Transit Center	58,000
5	Stagecoach St Andrews Depot	52,000

```
-- 10 This query gets information about bus drivers, their hourly salaries,
the total hours they have driven, and their total earnings based on the hours
driven.

SELECT
    s.staff_id AS 'Staff ID',
    s.staff_name AS 'Driver Name',
    d.hourly_salary_pounds AS 'Hourly Salary £',
    SUM(hd.hours_driven) AS 'Total Hours Driven',
    SUM(hd.hours_driven * d.hourly_salary_pounds) AS 'Total Earnings £'
FROM
    staff s
JOIN
    driver d ON s.staff_id = d.staff_id
LEFT JOIN
    hours_driven hd ON s.staff_id = hd.staff_id
GROUP BY
    s.staff_id, s.staff_name, d.hourly_salary_pounds, hd.staff_id;
```



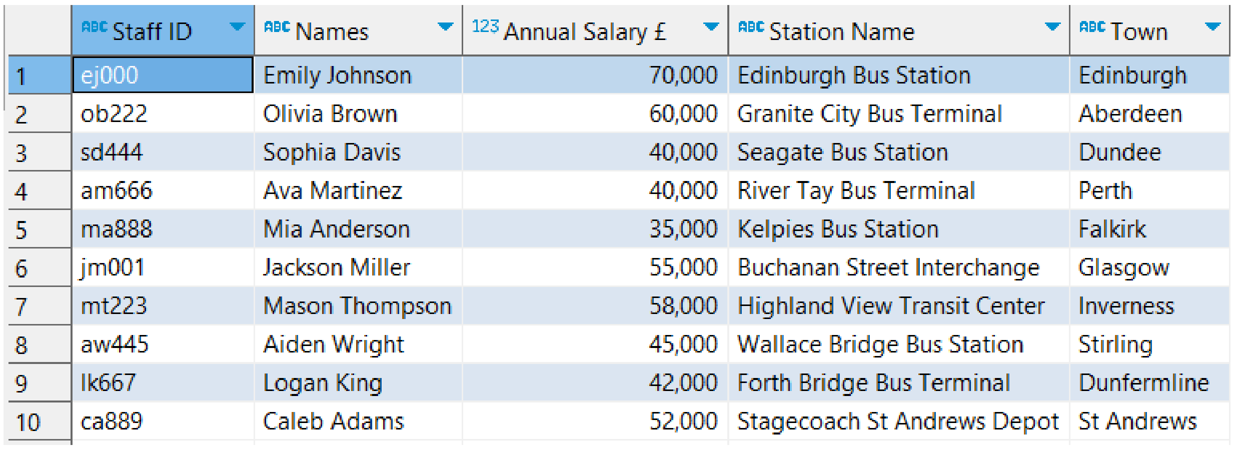
	Staff ID	Driver Name	Hourly Salary £	Total Hours Driven	Total Earnings £
1	ac778	Addison Carter	15	46	690
2	am990	Abigail Mitchell	18	45	810
3	bc777	Benjamin Clark	16	48	768
4	ch112	Charlotte Harris	17	38	646
5	ds111	Daniel Smith	20	46	920
6	ew555	Ethan Wilson	18	51	918
7	hh556	Harper Hall	16	40	640
8	lm333	Liam Miller	20	50	1,000
9	lr334	Lily Robinson	19	49	931
10	wt999	William Taylor	15	47	705

```
-- VIEWS
-- 1 Creating a view that shows information about top 3 managers that receive
the highest salaries among others.

CREATE VIEW top_3_salaries AS
SELECT staff_id AS 'Staff ID', annual_salary_pounds AS 'Annual Salary £',
town AS 'Town'
FROM manager NATURAL JOIN station
ORDER BY annual_salary_pounds DESC
LIMIT 3;

SELECT * FROM top_3_salaries;
```



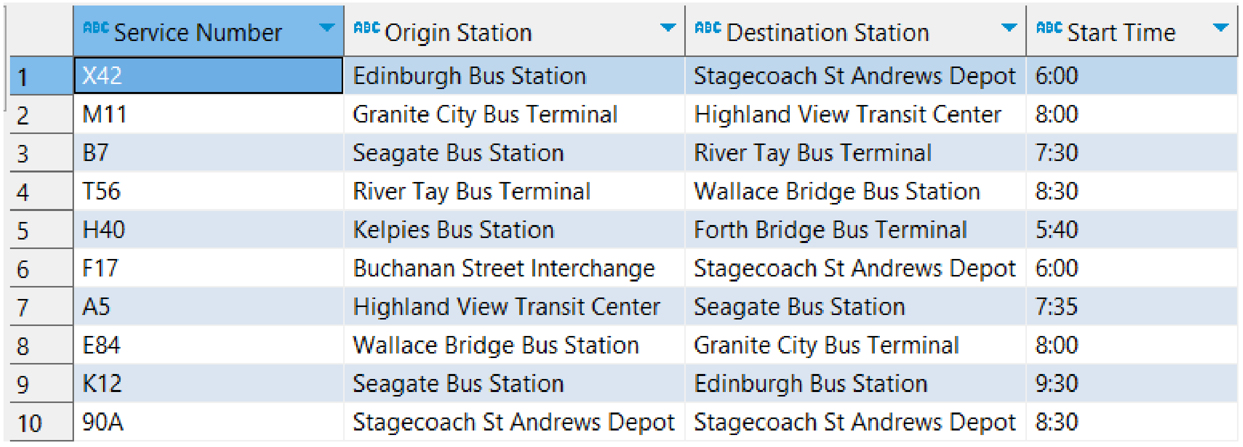
	Staff ID	Annual Salary £	Town
1	ej000	70,000	Edinburgh
2	ob222	60,000	Aberdeen
3	mt223	58,000	Inverness

```
-- 2 Creating a view that provides details about managers, their salary and
the stations they work at.

CREATE VIEW staff_manager_station_info AS
SELECT s.staff_id AS 'Staff ID', s.staff_name AS 'Names',
m.annual_salary_pounds AS 'Annual Salary £', st.station_name AS 'Station
Name', st.town AS 'Town'
FROM staff s
INNER JOIN manager m ON s.staff_id = m.staff_id
INNER JOIN station st ON s.staff_id = st.staff_id;

SELECT * FROM staff_manager_station_info;
```



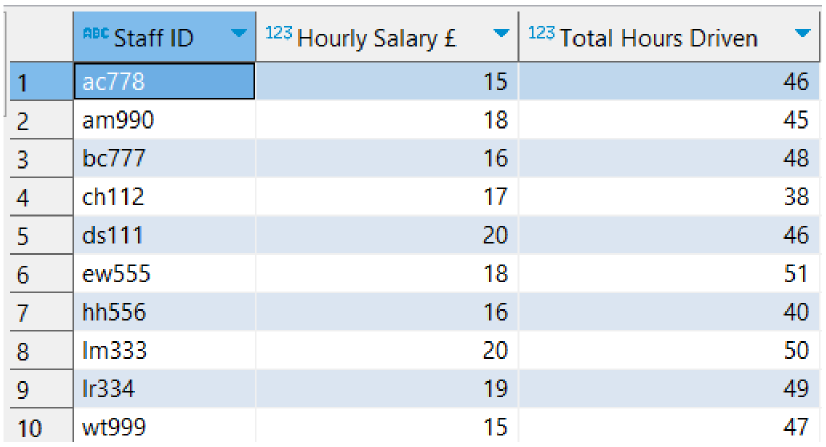
	Staff ID	Names	Annual Salary £	Station Name	Town
1	ej000	Emily Johnson	70,000	Edinburgh Bus Station	Edinburgh
2	ob222	Olivia Brown	60,000	Granite City Bus Terminal	Aberdeen
3	sd444	Sophia Davis	40,000	Seagate Bus Station	Dundee
4	am666	Ava Martinez	40,000	River Tay Bus Terminal	Perth
5	ma888	Mia Anderson	35,000	Kelpies Bus Station	Falkirk
6	jm001	Jackson Miller	55,000	Buchanan Street Interchange	Glasgow
7	mt223	Mason Thompson	58,000	Highland View Transit Center	Inverness
8	aw445	Aiden Wright	45,000	Wallace Bridge Bus Station	Stirling
9	lk667	Logan King	42,000	Forth Bridge Bus Terminal	Dunfermline
10	ca889	Caleb Adams	52,000	Stagecoach St Andrews Depot	St Andrews

```sql
-- 3 Creating a view that conveys the schedule for each bus service, the
service number, origin, destination, and the start time.

CREATE VIEW bus_schedule AS
SELECT s.service_number AS 'Service Number', s.origin_station AS 'Origin
Station', s.destination_station AS 'Destination Station', st.start_time AS
'Start Time'
FROM service s
JOIN service_time st ON s.service_number = st.service_number;

SELECT * FROM bus_schedule;
```



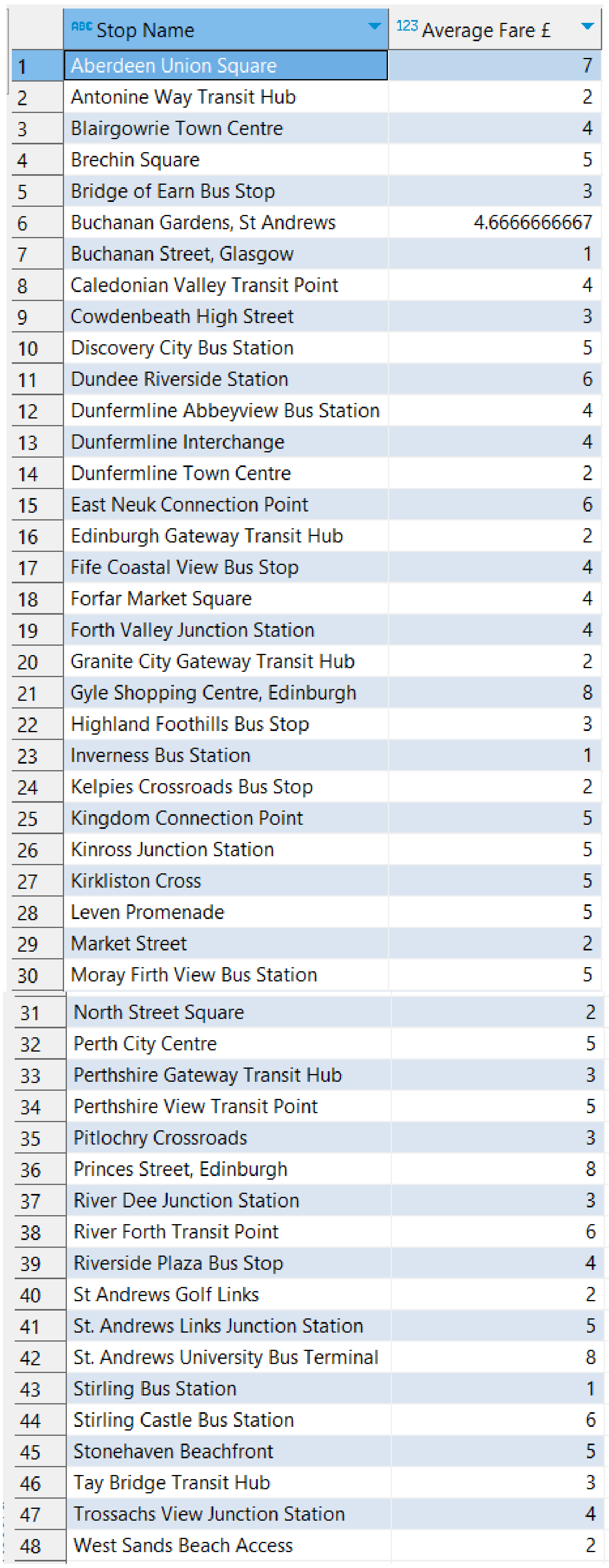
	Service Number	Origin Station	Destination Station	Start Time
1	X42	Edinburgh Bus Station	Stagecoach St Andrews Depot	6:00
2	M11	Granite City Bus Terminal	Highland View Transit Center	8:00
3	B7	Seagate Bus Station	River Tay Bus Terminal	7:30
4	T56	River Tay Bus Terminal	Wallace Bridge Bus Station	8:30
5	H40	Kelpies Bus Station	Forth Bridge Bus Terminal	5:40
6	F17	Buchanan Street Interchange	Stagecoach St Andrews Depot	6:00
7	A5	Highland View Transit Center	Seagate Bus Station	7:35
8	E84	Wallace Bridge Bus Station	Granite City Bus Terminal	8:00
9	K12	Seagate Bus Station	Edinburgh Bus Station	9:30
10	90A	Stagecoach St Andrews Depot	Stagecoach St Andrews Depot	8:30

```sql
-- 4 Creating a view that shows the total hours driven by each driver along
with their hourly salary.

CREATE VIEW driver_workload AS
SELECT d.staff_id AS 'Staff ID', d.hourly_salary_pounds AS 'Hourly Salary £',
SUM(hd.hours_driven) AS 'Total Hours Driven'
FROM driver d
LEFT JOIN hours_driven hd ON d.staff_id = hd.staff_id
GROUP BY d.staff_id, d.hourly_salary_pounds;

SELECT * FROM driver_workload;
```



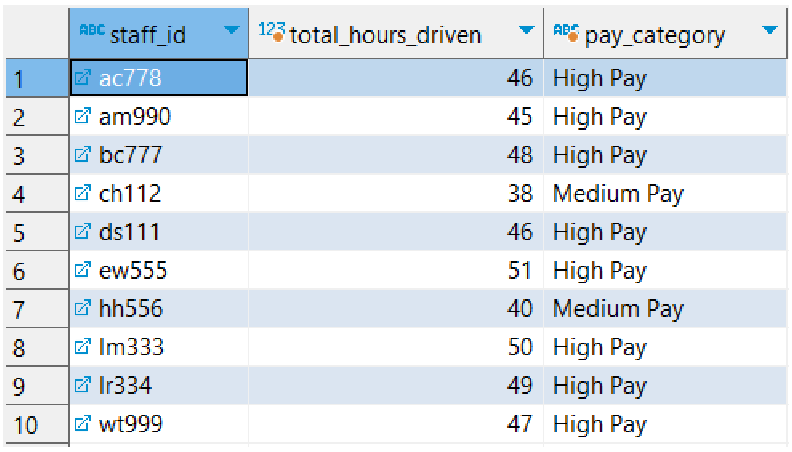
	Staff ID	Hourly Salary £	Total Hours Driven
1	ac778	15	46
2	am990	18	45
3	bc777	16	48
4	ch112	17	38
5	ds111	20	46
6	ew555	18	51
7	hh556	16	40
8	lm333	20	50
9	lr334	19	49
10	wt999	15	47

```sql
-- 5 Creating a view that shows the average fare for each bus stop based on
the fares from origin stations to each stop.

CREATE VIEW average_fare_by_stop AS
SELECT stop_name AS 'Stop Name', AVG(fare_pounds) AS 'Average Fare £'
FROM stop_on
GROUP BY stop_name;

SELECT * FROM average_fare_by_stop;
```



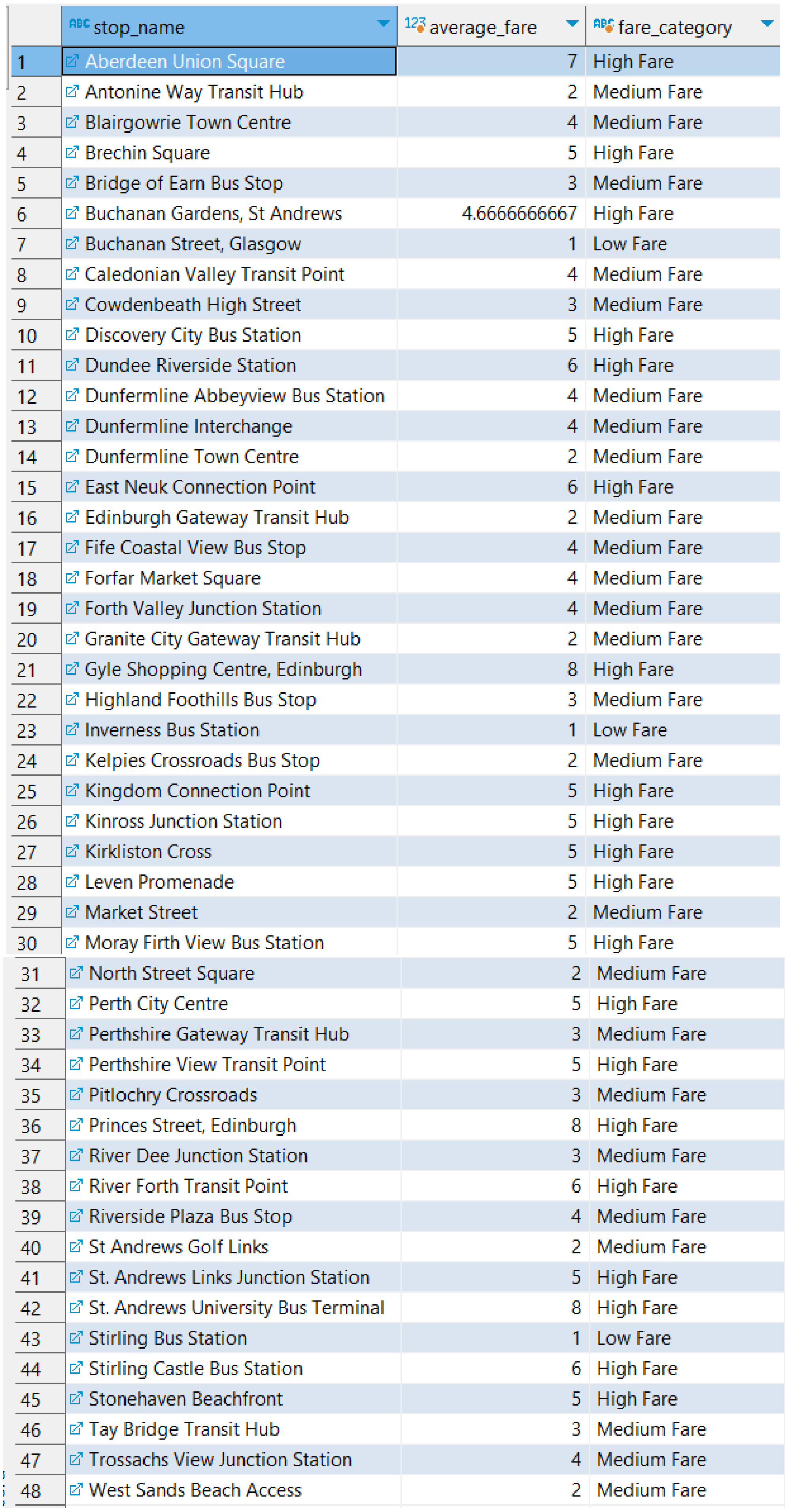
	Stop Name	Average Fare £
1	Aberdeen Union Square	7
2	Antonine Way Transit Hub	2
3	Blairgowrie Town Centre	4
4	Brechin Square	5
5	Bridge of Earn Bus Stop	3
6	Buchanan Gardens, St Andrews	4.6666666667
7	Buchanan Street, Glasgow	1
8	Caledonian Valley Transit Point	4
9	Cowdenbeath High Street	3
10	Discovery City Bus Station	5
11	Dundee Riverside Station	6
12	Dunfermline Abbeyview Bus Station	4
13	Dunfermline Interchange	4
14	Dunfermline Town Centre	2
15	East Neuk Connection Point	6
16	Edinburgh Gateway Transit Hub	2
17	Fife Coastal View Bus Stop	4
18	Forfar Market Square	4
19	Forth Valley Junction Station	4
20	Granite City Gateway Transit Hub	2
21	Gyle Shopping Centre, Edinburgh	8
22	Highland Foothills Bus Stop	3
23	Inverness Bus Station	1
24	Kelpies Crossroads Bus Stop	2
25	Kingdom Connection Point	5
26	Kinross Junction Station	5
27	Kirkliston Cross	5
28	Leven Promenade	5
29	Market Street	2
30	Moray Firth View Bus Station	5
31	North Street Square	2
32	Perth City Centre	5
33	Perthshire Gateway Transit Hub	3
34	Perthshire View Transit Point	5
35	Pitlochry Crossroads	3
36	Princes Street, Edinburgh	8
37	River Dee Junction Station	3
38	River Forth Transit Point	6
39	Riverside Plaza Bus Stop	4
40	St Andrews Golf Links	2
41	St. Andrews Links Junction Station	5
42	St. Andrews University Bus Terminal	8
43	Stirling Bus Station	1
44	Stirling Castle Bus Station	6
45	Stonehaven Beachfront	5
46	Tay Bridge Transit Hub	3
47	Trossachs View Junction Station	4
48	West Sands Beach Access	2

```
-- CASES (EXTRA)
-- 1 Creating a case statement to categorise drivers into different pay
categories based on the total hours they have driven.

SELECT
    staff_id,
    SUM(hours_driven) AS total_hours_driven,
    CASE
        WHEN SUM(hours_driven) < 20 THEN 'Low Pay'
        WHEN SUM(hours_driven) BETWEEN 20 AND 40 THEN 'Medium Pay'
        WHEN SUM(hours_driven) > 40 THEN 'High Pay'
    END AS pay_category
FROM hours_driven
GROUP BY staff_id;
```



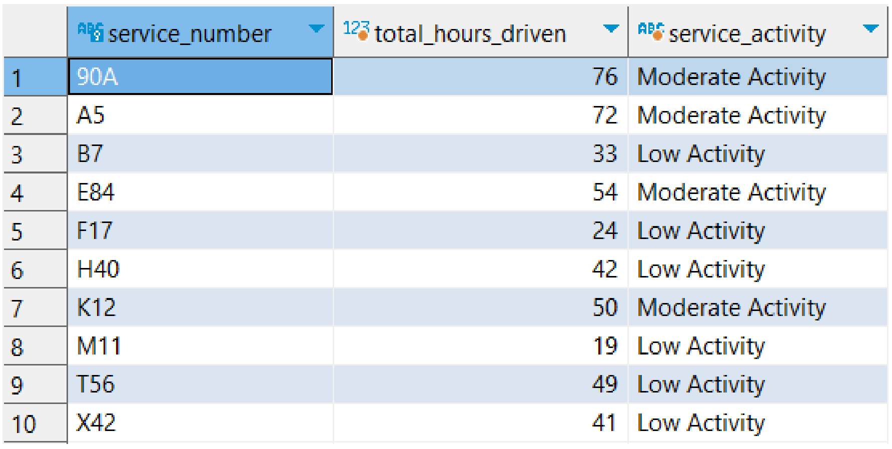
	staff_id	total_hours_driven	pay_category
1	ac778	46	High Pay
2	am990	45	High Pay
3	bc777	48	High Pay
4	ch112	38	Medium Pay
5	ds111	46	High Pay
6	ew555	51	High Pay
7	hh556	40	Medium Pay
8	lm333	50	High Pay
9	lr334	49	High Pay
10	wt999	47	High Pay

```
-- 2 Creating a case statement to categorise bus stops into different fare
levels.

SELECT
    stop_name,
    AVG(fare_pounds) AS average_fare,
    CASE
        WHEN AVG(fare_pounds) < 2 THEN 'Low Fare'
        WHEN AVG(fare_pounds) BETWEEN 2 AND 4 THEN 'Medium Fare'
        WHEN AVG(fare_pounds) > 4 THEN 'High Fare'
    END AS fare_category
FROM stop_on
GROUP BY stop_name;
```



	stop_name	average_fare	fare_category
1	Aberdeen Union Square	7	High Fare
2	Antonine Way Transit Hub	2	Medium Fare
3	Blairgowrie Town Centre	4	Medium Fare
4	Brechin Square	5	High Fare
5	Bridge of Earn Bus Stop	3	Medium Fare
6	Buchanan Gardens, St Andrews	4.6666666667	High Fare
7	Buchanan Street, Glasgow	1	Low Fare
8	Caledonian Valley Transit Point	4	Medium Fare
9	Cowdenbeath High Street	3	Medium Fare
10	Discovery City Bus Station	5	High Fare
11	Dundee Riverside Station	6	High Fare
12	Dunfermline Abbeyview Bus Station	4	Medium Fare
13	Dunfermline Interchange	4	Medium Fare
14	Dunfermline Town Centre	2	Medium Fare
15	East Neuk Connection Point	6	High Fare
16	Edinburgh Gateway Transit Hub	2	Medium Fare
17	Fife Coastal View Bus Stop	4	Medium Fare
18	Forfar Market Square	4	Medium Fare
19	Forth Valley Junction Station	4	Medium Fare
20	Granite City Gateway Transit Hub	2	Medium Fare
21	Gyle Shopping Centre, Edinburgh	8	High Fare
22	Highland Foothills Bus Stop	3	Medium Fare
23	Inverness Bus Station	1	Low Fare
24	Kelpies Crossroads Bus Stop	2	Medium Fare
25	Kingdom Connection Point	5	High Fare
26	Kinross Junction Station	5	High Fare
27	Kirkliston Cross	5	High Fare
28	Leven Promenade	5	High Fare
29	Market Street	2	Medium Fare
30	Moray Firth View Bus Station	5	High Fare
31	North Street Square	2	Medium Fare
32	Perth City Centre	5	High Fare
33	Perthshire Gateway Transit Hub	3	Medium Fare
34	Perthshire View Transit Point	5	High Fare
35	Pitlochry Crossroads	3	Medium Fare
36	Princes Street, Edinburgh	8	High Fare
37	River Dee Junction Station	3	Medium Fare
38	River Forth Transit Point	6	High Fare
39	Riverside Plaza Bus Stop	4	Medium Fare
40	St Andrews Golf Links	2	Medium Fare
41	St. Andrews Links Junction Station	5	High Fare
42	St. Andrews University Bus Terminal	8	High Fare
43	Stirling Bus Station	1	Low Fare
44	Stirling Castle Bus Station	6	High Fare
45	Stonehaven Beachfront	5	High Fare
46	Tay Bridge Transit Hub	3	Medium Fare
47	Trossachs View Junction Station	4	Medium Fare
48	West Sands Beach Access	2	Medium Fare

```sql
-- 3 Creating a case statement to categorise bus services into different
activity levels based on the total number of hours driven by drivers for each
service.

SELECT
    s.service_number,
    SUM(hd.hours_driven) AS total_hours_driven,
    CASE
        WHEN SUM(hd.hours_driven) < 50 THEN 'Low Activity'
        WHEN SUM(hd.hours_driven) BETWEEN 50 AND 100 THEN 'Moderate Activity'
        WHEN SUM(hd.hours_driven) > 100 THEN 'High Activity'
    END AS service_activity
FROM service s
LEFT JOIN hours_driven hd ON s.service_number = hd.service_number
GROUP BY s.service_number;
```



	service_number	total_hours_driven	service_activity
1	90A	76	Moderate Activity
2	A5	72	Moderate Activity
3	B7	33	Low Activity
4	E84	54	Moderate Activity
5	F17	24	Low Activity
6	H40	42	Low Activity
7	K12	50	Moderate Activity
8	M11	19	Low Activity
9	T56	49	Low Activity
10	X42	41	Low Activity

Task 4: Reflection

Reflecting on the process of converting the Entity-Relationship (E-R) model into SQL and working with data, I found the experience to be both rewarding and challenging.

What I Did Well

1. *Database Design*. I believe I successfully translated the requirements and the E-R model into a well-structured relational database design. Each table was created with careful consideration of its purpose and relationships were established effectively.

2. *Data Integrity*. Ensuring data integrity was a priority. By defining appropriate primary and foreign keys along with constraints, I could maintain consistency and accuracy within the database.

3. *Documentation*. I tried to maintain clarity and transparency in the SQL code by providing comments and documentation. This enhances readability and facilitates understanding the code and any modifications in future.

What Was Challenging

1. *Complex Relationships.* Managing complex relationships between entities, especially in scenarios involving multiple connections and dependencies was quite a challenge.

2. *Testing and Validation.* Ensuring that the SQL code worked as intended and validated data effectively was a continuous process. Identifying potential issues and validating the correctness of the queries took a lot of time.

What I Would Do Differently

Collaborative Approach. Collaborating with team members could provide additional insights and perspectives. Group work might lead to a more comprehensive database design.