

TITLE: COURSEWORK

NAME: JOSHUA ADDAI-MARNU

STUDENT NUMBER: 19548571

**COURSE: DATA SYSTEM CONCEPTS
AND FUNDAMENTALS**

DATE: 10/01/2023

PART B

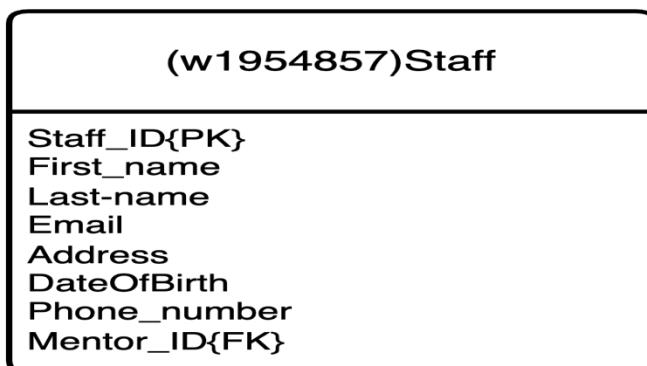
MAIN ENTITIES AND RELATIONSHIPS BETWEEN ENTITIES OF EXCEL UNIVERSITY DATABASE

❖ Staff mentors Staff

Staff(Mentor) mentors Staff(Mentee).

1. This is a recursive relationship(non-identifying relationship) between two entities, that is Mentor(Staff) and a Mentee(Staff).
2. Role names “Mentor(Staff) and Mentee(Staff)” were assigned to migrating foreign keys making the relationship to be Mentor(Staff) mentors Mentee(Staffs).
3. The parent table is the Mentor(Staff) entity and the child table is the Mentee(Staff).
4. In this particular relationship, the entity or table acting as the parent (Mentor) and the one serving as the child(Mentee) are the same.
5. The entity maintains a relationship with itself, where a key within it is continually moved or migrated.
6. Hence, the foreign key is “Mentor_ID”.

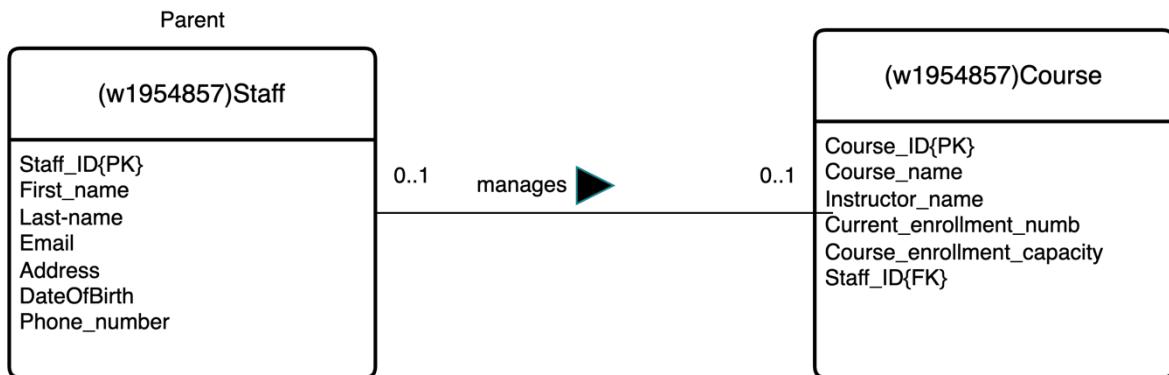
Logical ERD:



❖ Staff manages Course:

1. This is a one-to-one (1:1) relationship (optional on both sides).
2. Create two tables Two tables need to be created. That is, Staff and Course tables.
3. Staff table is made the Parent table since it is the stronger entity while the Course table is made the child table.
4. The foreign key of the Course table is the primary key of the Staff table.
5. Hence, the foreign key is Staff_ID.

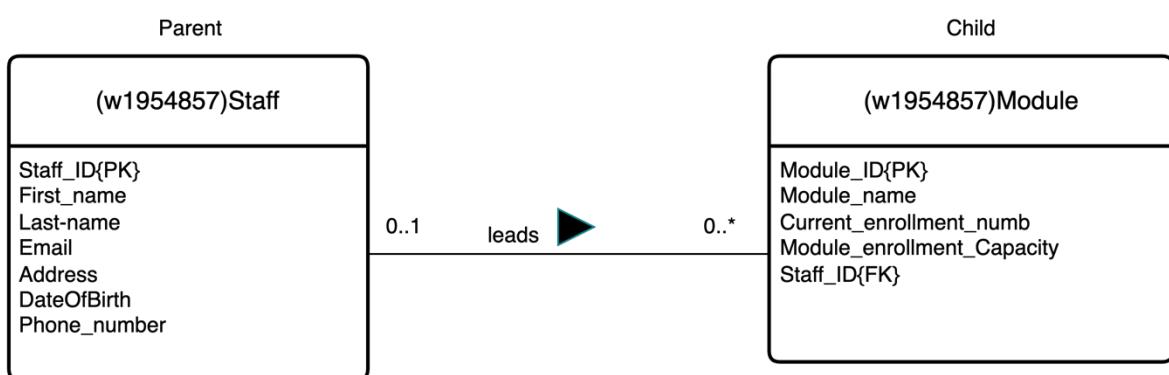
Logical ERD:



❖ Staff leads Module:

1. This is a one-to-many (1:M) relationship
2. Two tables are created namely the Staff entity and Module entity.
3. Parent entity is the entity that has a cardinality of “one”, that is Staff table and the Child entity is the entity that has a cardinality of “many”, that is Module table.
4. The foreign key is created on the Module table and it is the primary key of the staff entity.
5. Hence, the foreign key is Staff_ID which references the primary key of the Staff table.

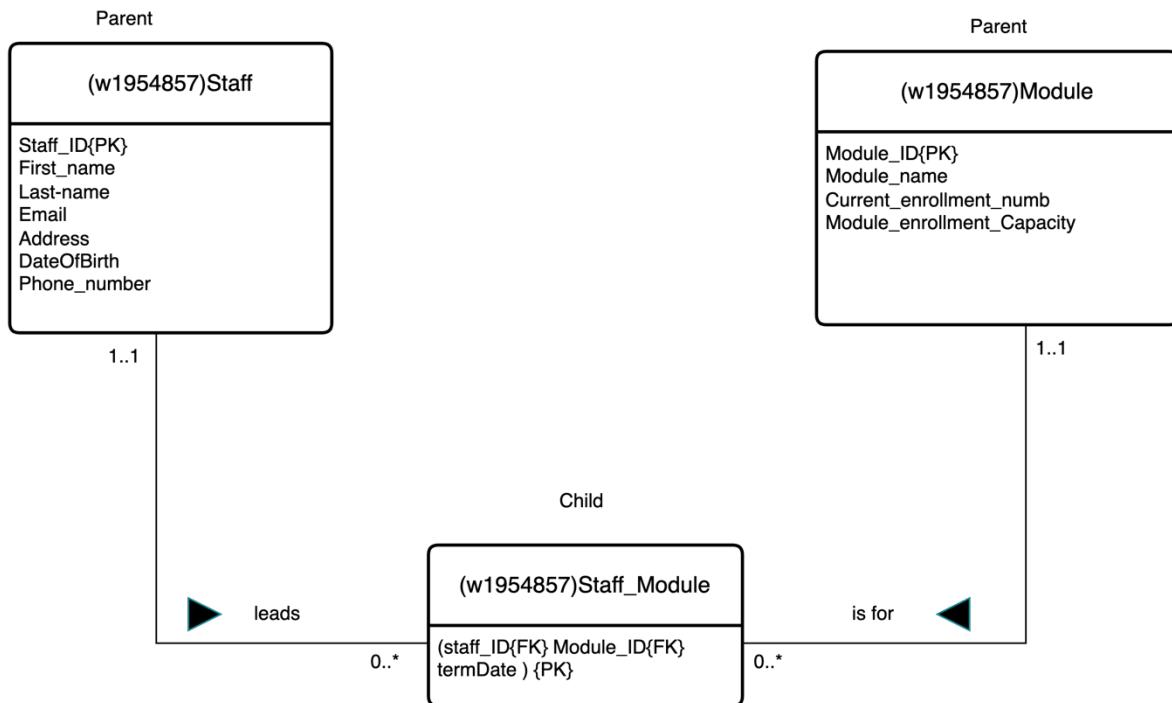
Logical ERD:



❖ **Staff delivers Module:**

1. This is a many-to-many (M:M) relationship.
2. Three tables are created, that is two Parent Tables and a link table as the Child table which is associated with the parent tables.
3. The two original parent tables are made of Staff and Module entities.
4. The link table is Child and is called the Staff_Module table.
5. Foreign keys of the Staff_Module table are the primary keys of both parent tables
6. Hence, the foreign keys are Staff_ID and Module_ID.
7. The primary key of the Staff_Module table is the combination of the two primary keys of Staff and Module entities.
8. Hence, the primary key of the link table is Staff_ID{FK}, Module{FK} making them a compound primary key as well.

Logical ERD:

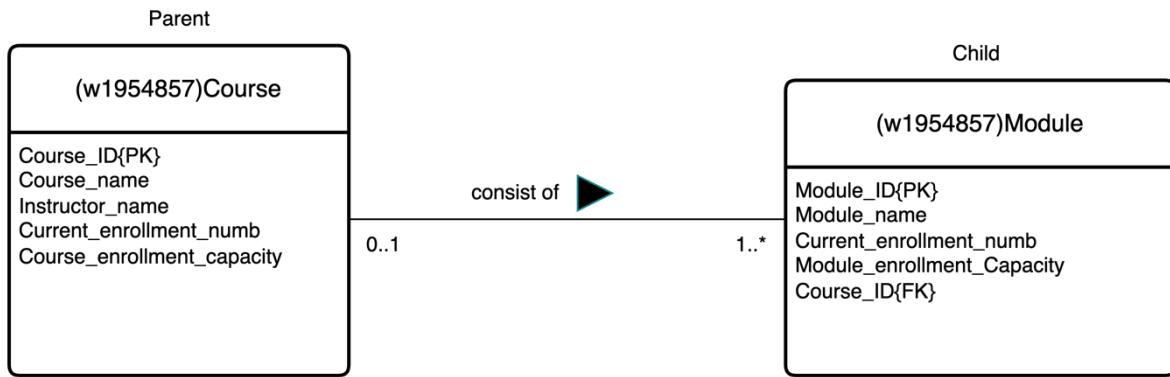


❖ **Course consists of Module:**

1. This is a one-to-many (1:M) relationship
2. Two tables are created namely the Course entity and Module entity.
3. Parent entity is the entity that has a cardinality of “one”, that is Course table and the Child entity is the entity that has a cardinality of “many”, that is Module table.
4. The foreign key is created on the Module table and it is the primary key of the Course entity.

5. Hence, the foreign key is Course_ID which references the primary key of the Course table.

Logical ERD:



❖ Student is registered on Course

1. This is a one-to-many (1:M) relationship
2. Two tables are created namely the Course entity and Student entity.
3. Parent entity is the entity that has a cardinality of “one”, that is Course table and the Child entity is the entity that has a cardinality of “many”, that is Student table.
4. The foreign key is created on the Student table and it is the primary key of the Course entity.
5. Hence, the foreign key is Course_ID which references the primary key of the Course table.

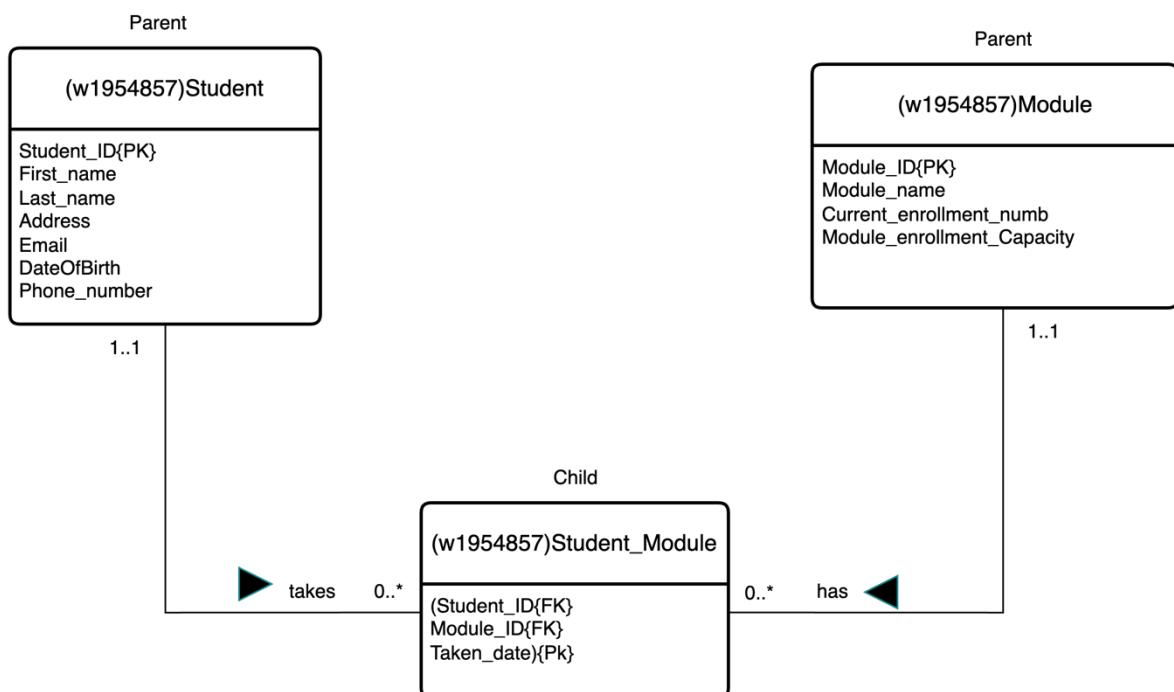
Logical ERD:



❖ **Student takes Module**

1. This is a many-to-many (M:M) relationship.
2. Three tables are created, that is two Parent Tables and a link table as the Child table which is associated with the parent tables.
3. The two original parent tables are made of Student and Module entities.
4. The link table is Child and is called the Student_Module table.
5. Foreign keys of the Student_Module table are the primary keys of both parent tables
6. Hence, the foreign keys are Staff_ID and Module_ID.
7. The primary key of the Student_Module table is the combination of the two primary keys of Staff and Module entities as well as an additional date since the combination of two primary keys can be repeated.
8. Hence, the primary key of the link table is Student_ID{FK}, Module{FK} and additional date which is “Taken_date” making it a composite primary key.

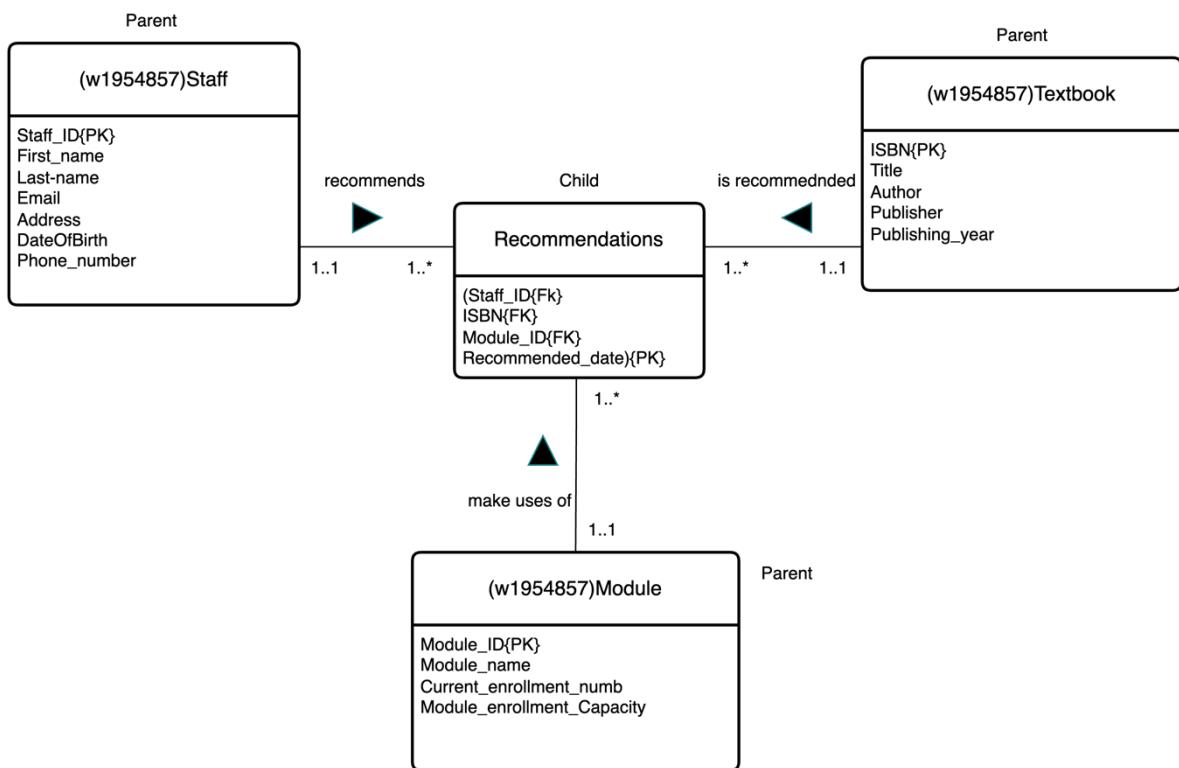
Logical ERD:



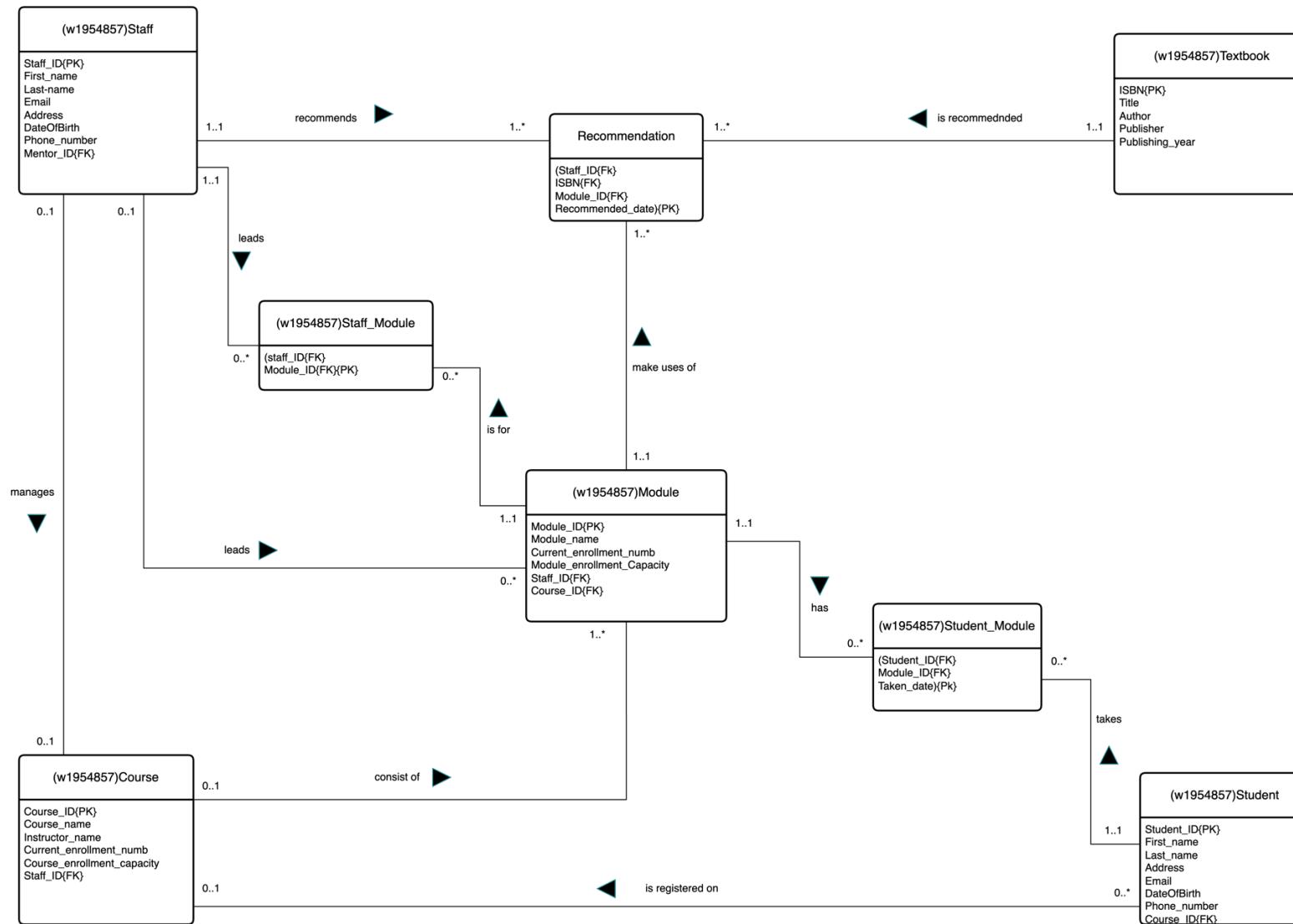
❖ **Lecturer(Staff) recommends a Textbook for each Module:**

1. This is a complex relationship(ternary).
2. Four tables are created, that is three Parent Tables and a link table as the Child table which is associated with the parent tables.
3. The three original parent tables are made of Staff, Module and Textbook entities.
4. The link table is Child and called the recommendation table.
5. The foreign keys of the Recommendation table are the primary keys of the parent tables
6. Hence, the foreign keys are Staff_ID, Module_ID and ISBN.
7. The primary key of the Recommendation table is the combination of the three primary keys of Staff, Module and Textbook entities as well as an additional date since the combination of three primary keys can be repeated.
8. Hence, the primary key of the link table is Staff_ID{FK}, Module{FK}, and the additional date is “Recommended_date” making it a composite primary key.

Logical ERD:



EXCEL University logical ERD:



PART A

THE MAIN ENTITIES OF THE DRIVING SCHOOL DATABASE:

1. Office
2. Staff
3. Instructor
4. Client
5. Vehicle
6. Lesson
7. Staff

THE MAIN RELATIONSHIPS BETWEEN THE ENTITIES(BINARY RELATIONSHIPS):

1. Office has Staff.
2. Instructor is a staff.
3. Client registers at office.
4. Instructor interview client.
5. Client book Lesson.
6. Instructor is allocated a vehicle.
7. Client takes Test.

QUATERNARY RELATIONSHIP:

8. Client attends a lesson with a particular instructor in a particular vehicle at a given time.

ATTRIBUTES AND MULTIPLICITIES FOR EACH RELATIONSHIP.

1. Office has Staff.
 - One office has a minimum of one staff.
 - One office has a maximum of many staff.
 - One staff belongs to a minimum of one office.
 - One staff belongs to a maximum of one office.
2. Instructor is a staff.
 - One instructor can be a minimum of one staff.
 - One instructor can be a maximum of one staff.
 - One staff is a minimum of zero instructors.
 - One staff member is a maximum of one instructor.
3. Client registers at office.
 - One Office registers a minimum of zero client
 - One office registers a maximum of many clients

- One client registers at a minimum of one office
- One client registers at a maximum of one office

4. Instructor interview client.

- One instructor interviews a minimum of zero clients.
- One instructor interviews a maximum of many clients.
- One client is interviewed by a minimum of one instructor.
- One client is interviewed by a maximum of one instructor.

5. Client book Lesson.

- One client books a minimum of one lesson.
- One client books a maximum of many lessons.
- One lesson is booked by a minimum of one client.
- One lesson is booked by a maximum of one client.

6. Instructor is allocated a vehicle.

- One instructor is allocated a minimum of one vehicle.
- One instructor is allocated a maximum of one vehicle.
- One vehicle is allocated to a minimum of zero instructors.
- One vehicle is allocated to a maximum of one instructor.

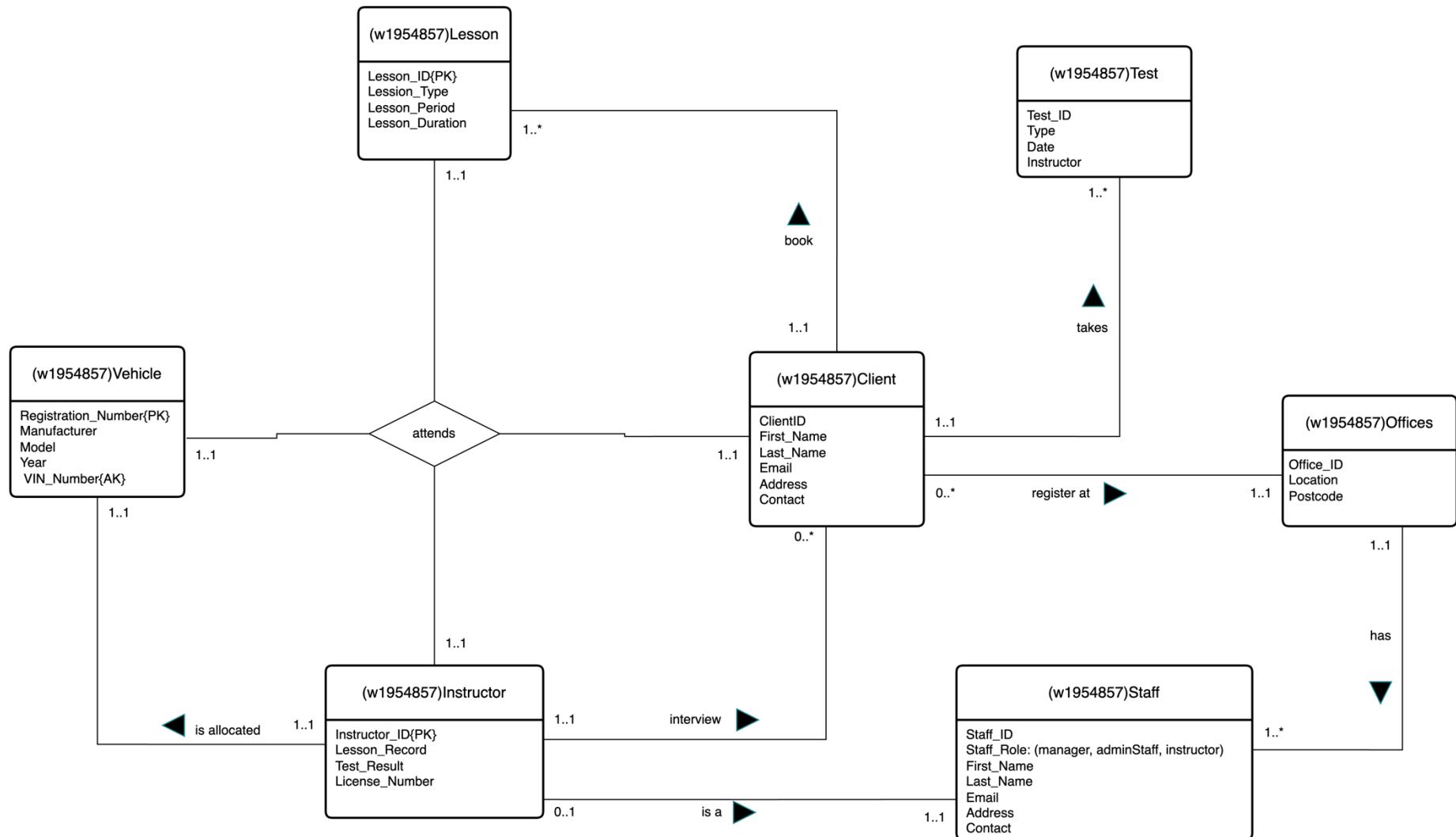
7. Client takes Test.

- One client takes a minimum of one test.
- One client takes a maximum of many tests.
- One test is taken by a minimum of one client.
- One test is taken by a maximum of one client.

8. Client attends a lesson with a particular instructor in a particular vehicle at a given time.

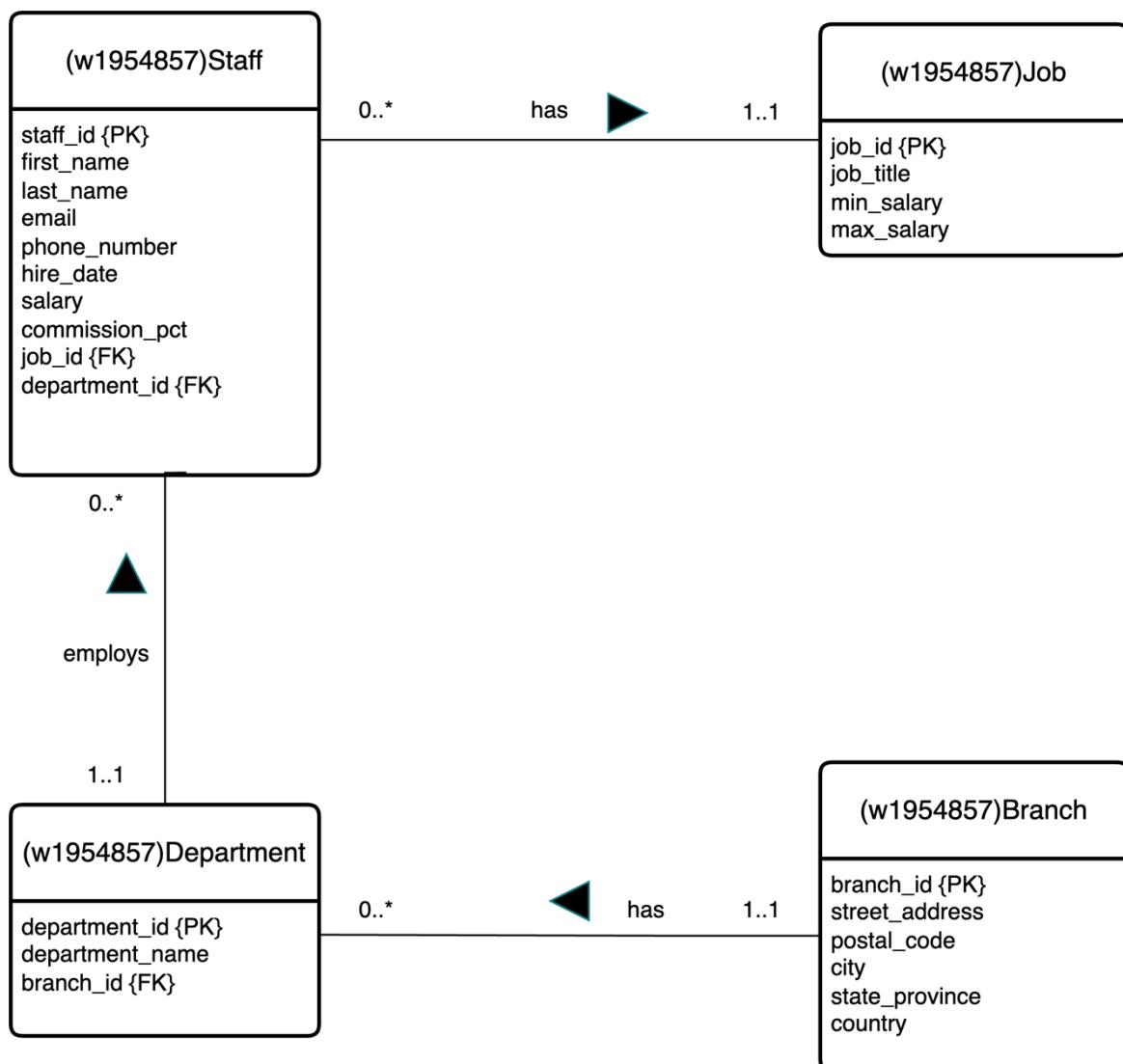
- Client attends a minimum of one lesson with a particular instructor in a particular vehicle.
- Client attends a maximum of one lesson with a particular instructor in a particular vehicle.
- Client attends a lesson with a minimum of one instructor in a particular vehicle.
- Client attends a lesson with a maximum of one instructor in a particular vehicle.
- Client attends a lesson with a particular instructor in a minimum of one vehicle.
- Client attends a lesson with a particular instructor in a maximum of one vehicle.
- A minimum of one client attends a lesson with a particular instructor in a particular vehicle.
- A minimum of one client attends a lesson with a particular instructor in a particular vehicle.

ERD representation of Driving School Database:



PART C

3.1 Modification to the ERD.



3.2. To create a department table, the branch table must be created first.

```
-- Creating branch table
```

```
CREATE TABLE w1954857_branch (
    branch_id VARCHAR(4) ,
    street_address VARCHAR(250) NOT NULL,
    postal_code VARCHAR(20) NOT NULL,
    city VARCHAR(100) NOT NULL,
    state_province VARCHAR(100) NOT NULL,
    country VARCHAR(100) NOT NULL,
    CONSTRAINT b_bid_pk PRIMARY KEY (branch_id)
);
```

The department table is created by referencing the branch table.

```
-- Creating the department table
```

```
CREATE TABLE w1954857_department (
    department_id VARCHAR(3),
    department_name VARCHAR(250) NOT NULL,
    branch_id VARCHAR(4),
    CONSTRAINT d_did_pk PRIMARY KEY (department_ID),
    CONSTRAINT d_bid_fk FOREIGN KEY (branch_id) REFERENCES
w1954857_branch(branch_id)
);
```

SCREENSHOT:

The screenshot shows the phpMyAdmin interface with the following details:

- URL:** phpmyadmin.ecs.westminster.ac.uk/index.php?route=/database/sql&db=w1954857_0
- Database:** w1954857_0
- Structure:** The left sidebar shows the database structure with tables: information_schema, performance_schema, w1954857_0 (which contains Branch, Job, Librarian, publisher, Staff, Userlib), w1954857_branch, and w1954857_department.
- SQL Tab:** The main area displays the SQL queries used to create the tables:

```
CREATE TABLE w1954857_branch ( branch_id VARCHAR(4) , street_address VARCHAR(250) NOT NULL, postal_code VARCHAR(20) NOT NULL, city VARCHAR(100) NOT NULL, state_province VARCHAR(100) NOT NULL, country VARCHAR(100) NOT NULL, CONSTRAINT b_bid_pk PRIMARY KEY (branch_id) );
CREATE TABLE w1954857_department ( department_id VARCHAR(3), department_name VARCHAR(250) NOT NULL, branch_id VARCHAR(4), CONSTRAINT d_did_pk PRIMARY KEY (department_ID), CONSTRAINT d_bid_fk FOREIGN KEY (branch_id) REFERENCES w1954857_branch(branch_id) );
```
- Messages:** Status messages indicate successful execution: "MySQL returned an empty result set (i.e. zero rows). (Query took 0.0708 seconds.)" and "MySQL returned an empty result set (i.e. zero rows). (Query took 0.0808 seconds.)".

3.3. Inserting content into the department table just created

- Inserting branches_id:

SQL CODE:

```
-- Inserting Branches in Various UK Cities
INSERT INTO w2025844_branch (branch_id, street_address, postal_code, city,
state_province, country)
VALUES
    ('B100', '101 Abbeywood Street', 'MA8 4AB', 'Manchester', 'Greater Manchester',
'United Kingdom'),
    ('B200', '202 Loampit Road', 'DA8 1FH', 'London', 'Greater London', 'United
Kingdom'),
    ('B300', '303 Ilford Avenue', 'BC2 9EY', 'Birmingham', 'West Midlands', 'United
Kingdom'),
    ('B400', '404 Dartford Street', 'EF9 6XG', 'Edinburgh', 'Scotland', 'United
Kingdom');
```

- SCREENSHOT OF STRUCTURE OF THE BRANCH TABLE:

The screenshot shows the phpMyAdmin interface for the database 'w1954857_0'. The left sidebar shows the database structure with tables: 'information_schema', 'performance_schema', 'w1954857_0' (which contains 'New', 'Book', 'Branch', 'Job', 'Librarian', 'publisher', 'Staff', 'Userlib', 'w1954857_branch', and 'w1954857_department'), and 'w1954857_branch'. The main panel displays the 'w1954857_branch' table. The table has columns: branch_id, street_address, postal_code, city, state_province, and country. The data shows four rows of branch information:

branch_id	street_address	postal_code	city	state_province	country
B100	101 Abbeywood Street	MA8 4AB	Manchester	Greater Manchester	United Kingdom
B200	202 Loampit Road	DA8 1FH	London	Greater London	United Kingdom
B300	303 Ilford Avenue	BC2 9EY	Birmingham	West Midlands	United Kingdom
B400	404 Dartford Street	EF9 6XG	Edinburgh	Scotland	United Kingdom

Below the table, there are buttons for 'Check all', 'With selected:', and 'Export'. At the bottom, there are links for 'Print', 'Copy to clipboard', 'Export', 'Display chart', and 'Create view'.

- Inserting departments:

SQL CODE:

-- Inserting Departments

```
INSERT INTO w2025844_department (department_ID, department_name, branch_id)
VALUES
    ('D10', 'IT', 'B100'),
    ('D20', 'Admin', 'B200'),
    ('D30', 'Sales', 'B300'),
    ('D40', 'Marketing', 'B400'),
    ('D50', 'Customer Care', 'B300');
```

- SCREENSHOT OF THE STRUCTURE OF DEPARTMENT TABLE:

The screenshot shows the phpMyAdmin interface for the 'w1954857_0' database. The left sidebar lists databases and tables, with 'w1954857_branch' and 'w1954857_department' selected. The main panel displays the 'Structure' tab for the 'w1954857_department' table. The SQL query shown is 'SELECT * FROM `w1954857_department`'. The results table shows five rows of data:

	department_id	department_name	branch_id
<input type="checkbox"/>	D10	IT	B100
<input type="checkbox"/>	D20	Admin	B200
<input type="checkbox"/>	D30	Sales	B300
<input type="checkbox"/>	D40	Marketing	B400
<input type="checkbox"/>	D50	Customer Care	B300

Below the table are buttons for 'Check all', 'With selected:', and 'Edit', 'Copy', 'Delete', 'Export'.

3.4. To create a query to display/list,

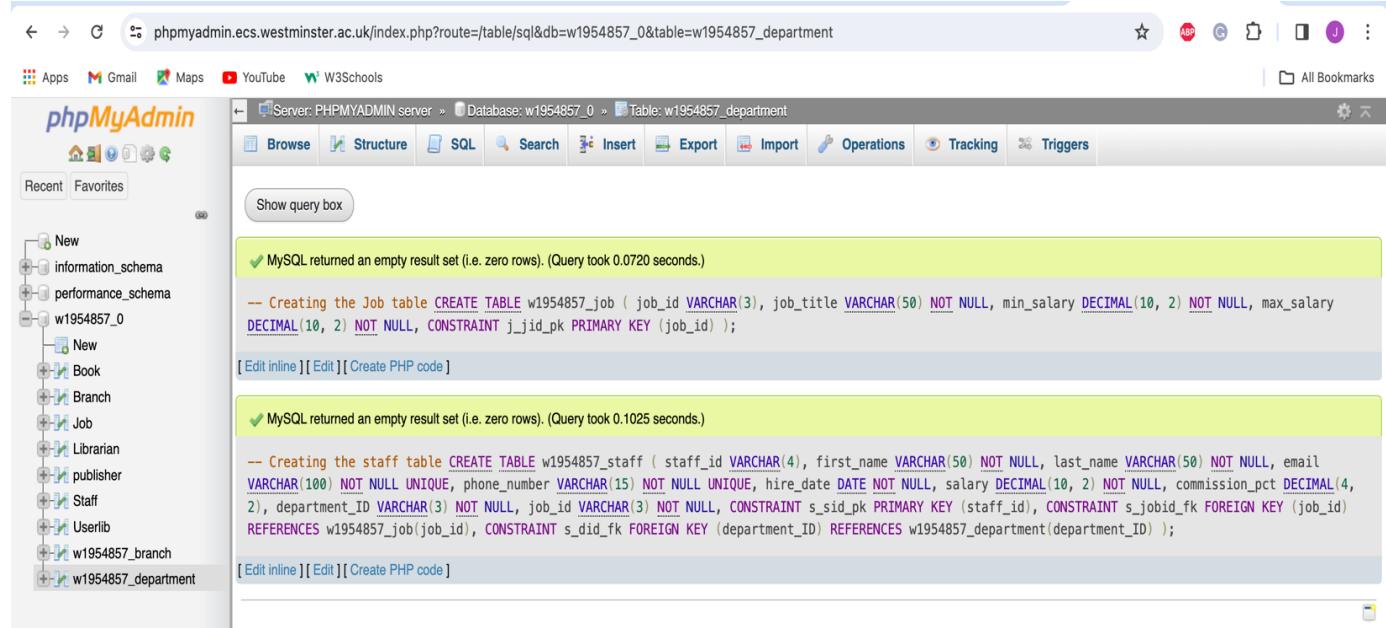
- I. The job and staff table must be created.

SQL CODE:

```
-- Creating the Job table
CREATE TABLE w1954857_job (
    job_id VARCHAR(3),
    job_title VARCHAR(50) NOT NULL,
    min_salary DECIMAL(10, 2) NOT NULL,
    max_salary DECIMAL(10, 2) NOT NULL,
    CONSTRAINT j_jid_pk PRIMARY KEY (job_id)
);

-- Creating the staff table
CREATE TABLE w1954857_staff (
    staff_id VARCHAR(4),
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    phone_number VARCHAR(15) NOT NULL UNIQUE,
    hire_date DATE NOT NULL,
    salary DECIMAL(10, 2) NOT NULL,
    commission_pct DECIMAL(4, 2),
    department_ID VARCHAR(3) NOT NULL,
    job_id VARCHAR(3) NOT NULL,
    CONSTRAINT s_sid_pk PRIMARY KEY (staff_id),
    CONSTRAINT s_jobid_fk FOREIGN KEY (job_id) REFERENCES w1954857_job(job_id),
    CONSTRAINT s_did_fk FOREIGN KEY (department_ID) REFERENCES
w1954857_department(department_ID)
);
```

Screenshot:



II. Inserting data into the job table.

SQL CODE:

```
-- Inserting data into job table
INSERT INTO w1954857_job (job_id, job_title, min_salary, max_salary)
VALUES
    ('901', 'Managing Director', 75000, 125000),
    ('902', 'Clerical Administrator', 28000, 41000),
    ('903', 'System Analyst', 45000, 95000),
    ('904', 'Security Officer', 30000, 55000),
    ('905', 'Network Operator', 37000, 68000),
    ('906', 'Data Entry Operator', 24000, 50000),
    ('907', 'Insurance Rep', 30000, 60000);
```

- SCRENSHOT OF STRUCTURE OF THE JOB TABLE:

The screenshot shows the phpMyAdmin interface for a MySQL database named 'w1954857_0'. The left sidebar displays the database schema with several tables listed under the 'w1954857_0' database. The 'w1954857_job' table is currently selected and highlighted in grey. The main panel shows the structure of the 'w1954857_job' table, which has four columns: 'job_id', 'job_title', 'min_salary', and 'max_salary'. Below the structure, a list of 7 rows is displayed, each corresponding to one of the SQL INSERT statements shown in the code block above. The rows are numbered 901 through 907. Each row includes edit, copy, delete, and export options. The top navigation bar shows the URL as 'phpmyadmin.ecs.westminster.ac.uk/index.php?route=/sql&db=w1954857_0&table=w1954857_job&pos=0'.

job_id	job_title	min_salary	max_salary
901	Managing Director	75000.00	125000.00
902	Clerical Administrator	28000.00	41000.00
903	System Analyst	45000.00	95000.00
904	Security Officer	30000.00	55000.00
905	Network Operator	37000.00	68000.00
906	Data Entry Operator	24000.00	50000.00
907	Insurance Rep	30000.00	60000.00

III. Inserting into staff table:

SQL CODE:

```
-- Inserting data into the staff table
INSERT INTO w1954857_staff (staff_id, first_name, last_name, email, phone_number,
hire_date, salary, commission_pct, job_id, department_ID)
VALUES
    (1001, 'Jim', 'King', 'jk@firm.com', '02079111001', '2011-01-21', 98000, NULL,
901, 'D10'),
    (1002, 'Jane', 'Queen', 'jq@firm.com', '02079111002', '2012-02-05', 99000,
NULL, 901, 'D20'),
    (1003, 'Jen', 'Probert', 'jp@firm.com', '02079111003', '2014-11-23', 79000,
NULL, 903, 'D10'),
    (1004, 'Mike', 'Brent', 'mb@firm.com', '02079111004', '2013-10-06', 51000,
NULL, 904, 'D20'),
    (1005, 'Nadia', 'Tamsa', 'nt@firm.com', '02079111005', '2013-10-08', 62000,
NULL, 905, 'D30'),
    (1006, 'Mo', 'Ali', 'ma@firm.com', '02079111006', '2015-11-24', 41000, 0.15,
907, 'D40'),
    (1007, 'Dannie', 'Kolova', 'dk@firm.com', '02079111007', '2016-05-15', 38000,
0.25, 907, 'D10'),
    (1008, 'Manu', 'Ogoda', 'mo@firm.com', '02079111008', '2017-08-12', 33000,
0.35, 907, 'D20'),
    (1009, 'Marc', 'Daniel', 'md@firm.com', '02079111009', '2014-01-02', 35000,
0.35, 907, 'D30'),
    (1010, 'Louise', 'Matos', 'lm@firm.com', '02079111010', '2017-11-05', 53000,
NULL, 905, 'D20'),
    (1011, 'Ram', 'Binghi', 'rb@firm.com', '02079111011', '2012-03-30', 35000,
NULL, 906, 'D10'),
    (1012, 'Tim', 'Norm', 'tn@firm.com', '02079111012', '2018-03-31', 48000, NULL,
906, 'D30'),
    (1013, 'Alex', 'Smart', 'as@firm.com', '02079111013', '2012-03-30', 39000,
NULL, 905, 'D10'),
    (1014, 'Bruno', 'Silba', 'bs@firm.com', '02079111014', '2014-05-08', 37000,
NULL, 905, 'D40'),
    (1015, 'Laurie', 'Kaldav', 'lk@firm.com', '02079111015', '2017-08-11', 34000,
NULL, 902, 'D10'),
    (1016, 'Sophie', 'Lanou', 'sl@firm.com', '02079111016', '2017-08-19', 34000,
NULL, 902, 'D20'),
    (1017, 'Yann', 'Taylor', 'yt@firm.com', '02079111017', '2018-09-03', 44000,
NULL, 904, 'D30'),
```

```

(1018, 'Sam', 'Tring', 'st@firm.com', '02079111018', '2018-09-05', 47000, NULL,
903, 'D20'),
(1019, 'Don', 'Matos', 'dt@firm.com', '02079111019', '2017-10-04', 49000, NULL,
903, 'D30'),
(1020, 'Dan', 'Mitch', 'dm@firm.com', '02079111020', '2019-01-14', 35000, NULL,
902, 'D30'),
(1021, 'Jean', 'Novski', 'jn@firm.com', '02079111021', '2019-02-13', 38000,
NULL, 906, 'D20'),
(1022, 'Malia', 'Mundi', 'mm@firm.com', '02079111022', '2019-07-15', 43000,
NULL, 906, 'D30'),
(1023, 'Kurt', 'Thorpe', 'kt@firm.com', '02079111023', '2019-07-15', 44000,
NULL, 906, 'D40');

```

SCREENSHOT OF STRUCTURE OF THE STAFF TABLE :

The screenshot shows the phpMyAdmin interface for the 'w1954857_0' database. The left sidebar shows the database structure with tables: information_schema, performance_schema, w1954857_0, and w1954857_staff. The w1954857_staff table is selected. The main area displays the table structure with the following columns:

staff_id	first_name	last_name	email	phone_number	hire_date	salary	commission_pct	department_ID	job_id
1001	Jim	King	jk@firm.com	02079111001	2011-01-21	98000.00	NULL	D10	901
1002	Jane	Queen	jq@firm.com	02079111002	2012-02-05	99000.00	NULL	D20	901
1003	Jen	Probert	jp@firm.com	02079111003	2014-11-23	79000.00	NULL	D10	903
1004	Mike	Brent	mb@firm.com	02079111004	2013-10-06	51000.00	NULL	D20	904
1005	Nadia	Tamsa	nt@firm.com	02079111005	2013-10-08	62000.00	NULL	D30	905
1006	Mo	Ali	ma@firm.com	02079111006	2015-11-24	41000.00	0.15	D40	907
1007	Dannie	Kolova	dk@firm.com	02079111007	2016-05-15	38000.00	0.25	D10	907
1008	Manu	Ogoda	mo@firm.com	02079111008	2017-08-12	33000.00	0.35	D20	907
1009	Marc	Daniel	md@firm.com	02079111009	2014-01-02	35000.00	0.35	D30	907
1010	Louise	Matos	lm@firm.com	02079111010	2017-11-05	53000.00	NULL	D20	905
1011	Ram	Binghi	rb@firm.com	02079111011	2012-03-30	35000.00	NULL	D10	906
1012	Tim	Norm	tn@firm.com	02079111012	2018-03-31	48000.00	NULL	D30	906
1013	Alex	Smart	as@firm.com	02079111013	2012-03-30	39000.00	NULL	D10	905
1014	Bruno	Silba	bs@firm.com	02079111014	2014-05-08	37000.00	NULL	D40	905
1015	Laurie	Kaldav	lk@firm.com	02079111015	2017-08-11	34000.00	NULL	D10	902
1016	Sophie	Lanou	sl@firm.com	02079111016	2017-08-19	34000.00	NULL	D20	902
1017	Yann	Taylor	yt@firm.com	02079111017	2018-09-03	44000.00	NULL	D30	904
1018	Sam	Tring	st@firm.com	02079111018	2018-09-05	47000.00	NULL	D20	903
1019	Don	Matos	dt@firm.com	02079111019	2017-10-04	49000.00	NULL	D30	903
1020	Dan	Mitch	dm@firm.com	02079111020	2019-01-14	35000.00	NULL	D30	902
1021	Jean	Novski	jn@firm.com	02079111021	2019-02-13	38000.00	NULL	D20	906
1022	Malia	Mundi	mm@firm.com	02079111022	2019-07-15	43000.00	NULL	D30	906
1023	Kurt	Thorpe	kt@firm.com	02079111023	2019-07-15	44000.00	NULL	D40	906

- a) An SQL query to display the branch id, street address and postal code for all the branches situated under the postcode starting from “B”. Ordered in ascending order by branch id.

SQL CODE:

```
SELECT branch_id,street_address,postal_code
FROM w1954857_branch
WHERE postal_code LIKE 'B%'
ORDER BY branch_id ASC;
```

Screenshot of query:

The screenshot shows the phpMyAdmin interface for a database named 'w1954857_0'. The left sidebar lists various schemas and tables, with 'w1954857_branch' selected. The main area displays the results of the executed SQL query:

```
SELECT branch_id,street_address,postal_code FROM w1954857_branch WHERE postal_code LIKE 'B%' ORDER BY branch_id ASC;
```

The results show one row:

branch_id	street_address	postal_code
B300	303 Ilford Avenue	BC2 9EY

Below the table, there are 'Query results operations' buttons for Print, Copy to clipboard, Export, Display chart, and Create view.

- b) The job id, job title and maximum salary of employees who might earn a maximum salary of 50000 and above and is working as an operator.

SQL CODE:

```
SELECT Job_id,job_title,max_salary
FROM w1954857_job
WHERE (max_salary >= 50000) AND (job_title LIKE '%operator%');
```

Screenshot of query:

The screenshot shows the phpMyAdmin interface on a web browser. The URL in the address bar is `phpmyadmin.ecs.westminster.ac.uk/index.php?route=/table/sql&db=w1954857_0&table=w1954857_job`. The left sidebar lists databases and tables, with 'w1954857_0' selected. The main area shows the results of the executed SQL query:

```
SELECT Job_id,job_title,max_salary FROM w1954857_job WHERE (max_salary >= 50000) AND (job_title LIKE '%operator%');
```

The results table displays two rows:

Job_id	job_title	max_salary
905	Network Operator	68000.00
906	Data Entry Operator	50000.00

Below the table are standard MySQL operations: Print, Copy to clipboard, Export, Display chart, and Create view.

- c) The last name, job id, hire date and salary of employees who work in branch B200 and who earns less than 50000 and greater than 60000, as well as those who work in the same branch and were hired before the 15th August 2017.

SQL CODE:

```
SELECT s.last_name, s.job_id, s.hire_date, s.salary
FROM w1954857_staff s
JOIN w1954857_department d ON s.department_ID = d.department_ID
WHERE (d.branch_id = 'B200' AND (s.salary < 50000 OR s.salary > 60000))
    OR (d.branch_id = 'B200' AND s.hire_date < '2017-08-15');
```

Screenshot of query:

The screenshot shows the phpMyAdmin interface with the following details:

- URL:** `phpmyadmin.ecs.westminster.ac.uk/index.php?route=/table/sql&db=w1954857_0&table=w1954857_staff`
- Database:** `w1954857_0`
- Table:** `w1954857_staff`
- Query:**

```
SELECT s.last_name, s.job_id, s.hire_date, s.salary
FROM w1954857_staff s
JOIN w1954857_department d ON s.department_ID = d.department_ID
WHERE (d.branch_id = 'B200' AND (s.salary < 50000 OR s.salary > 60000))
    OR (d.branch_id = 'B200' AND s.hire_date < '2017-08-15');
```
- Results:** 6 rows are shown, all of which have the same values as the original screenshot.

last_name	job_id	hire_date	salary
Queen	901	2012-02-05	99000.00
Brent	904	2013-10-06	51000.00
Ogoda	907	2017-08-12	33000.00
Lanou	902	2017-08-19	34000.00
Tring	903	2018-09-01	47000.00
Novski	906	2019-02-13	38000.00

- d) The last name, job, and salary for all employees whose job id is 907 and whose salary is not equal to £33,000 or £53,000.

SQL CODE:

```
e) SELECT last_name, job_id, salary  
f) FROM w1954857_staff  
g) WHERE job_id = '907'  
h) AND salary NOT IN (33000, 53000);
```

Screenshot:

The screenshot shows the phpMyAdmin interface. On the left, the database structure is visible with the 'w1954857_0' schema selected. The 'w1954857_staff' table is currently being viewed. The main area displays the results of the following SQL query:

```
SELECT last_name, job_id, salary FROM w1954857_staff WHERE job_id = '907' AND salary NOT IN (33000, 53000);
```

The results show three rows:

	last_name	job_id	salary			
<input type="checkbox"/>	Edit	Copy	Delete	All	907	41000.00
<input type="checkbox"/>	Edit	Copy	Delete	Kolova	907	38000.00
<input type="checkbox"/>	Edit	Copy	Delete	Daniel	907	35000.00

At the bottom, there are various operations like Print, Copy to clipboard, Export, Display chart, and Create view.

- e) The minimum and maximum salary of Managing Director. Label the columns as Salary Range MIN and Salary Range MAX.

```
SELECT MIN(min_salary) AS "Salary Range Min", MAX(max_salary) AS "Salary Range Max"
FROM w1954857_job
WHERE job_title = 'Managing Director';
```

The screenshot shows the phpMyAdmin interface. The left sidebar displays a tree view of databases and tables, with 'w1954857_0' selected. The main area shows the results of the executed SQL query:

```
SELECT MIN(min_salary) AS "Salary Range Min", MAX(max_salary) AS "Salary Range Max"
FROM w1954857_job
WHERE job_title = 'Managing Director';
```

The results table has two columns: 'Salary Range Min' and 'Salary Range Max'. The data row shows 75000.00 and 125000.00 respectively. Below the table are various operations like Print, Copy to clipboard, Export, Display chart, and Create view.