



INFORMATICS
INSTITUTE OF
TECHNOLOGY

UNIVERSITY OF
WESTMINSTER

INFORMATICS INSTITUTE OF TECHNOLOGY

COMPUTER SCIENCE PRACTICE
4COSC008C.2

SEMESTER 2 → COURSEWORK 2
DATA STORAGE AND MANUPULATION

Module Leader's Name - Ms. Sulochana Rupasinghe

Name - Ashfaaq Ahamed Hilal

UoW No. – 17613343/1

Student ID - 2019394

TABLE OF CONTENTS

1. INTRODUCTION	4
2. CONCEPTUAL DIAGRAM.....	5
1.1. Assumptions taken when implementing the CONCEPTUAL DIAGRAM. 6	
1.1.1. Assumptions about Entities	6
1.1.2. Assumptions about attributes.....	7
1.1.3. Assumptions about Primary keys	8
1.1.4. Assumptions about relationships	9
1.1.5. Assumptions about participations and cardinality.....	10
3. LOGICAL DIAGRAM	11
3.1. Assumptions taken when implementing the LOGICAL DIAGRAM.	12
3.1.1. Assumptions about Entities	12
3.1.2. Assumptions about attributes.....	13
3.1.3. Assumptions about participations and cardinality.....	14
3.1.4. Assumptions about the data type.....	15
4. CREATING TABLES.....	17
4.1. Client table.....	17
4.2. Course table	18
4.3. Client_Course table	19
4.4. Booking table	20
4.5. Package table	21
4.6. Package_Booking table	22
4.7. Expert table.....	23
4.8. Expert_Package table.....	24
4.9. Employee table	25
4.10. Expert_Employee table.....	26
5. INSERTING TABLES.....	27
5.1. Client Table	27
5.2. Course table	28
5.3. Client_Course table	29
5.4. Booking table	30

5.5. Package table	31
5.6. Package_Booking table	32
5.7. Expert table.....	33
5.8. Expert_Package table.....	34
5.9. Employee table	35
5.10. Expert_Employee table.....	36
6. Select Queries	37
6.1. Query 1	37
6.2. Query 2	38
6.3. Query 3	39
6.4. Query 4	40
6.5. Query 5	41
6.6. Query 6	42
6.7. Query 7	43
6.8. Query 8	44
7. References.....	45

1.INTRODUCTION

Database is a well-structured and organized method of storing data in a system. It can be accessed in various ways.

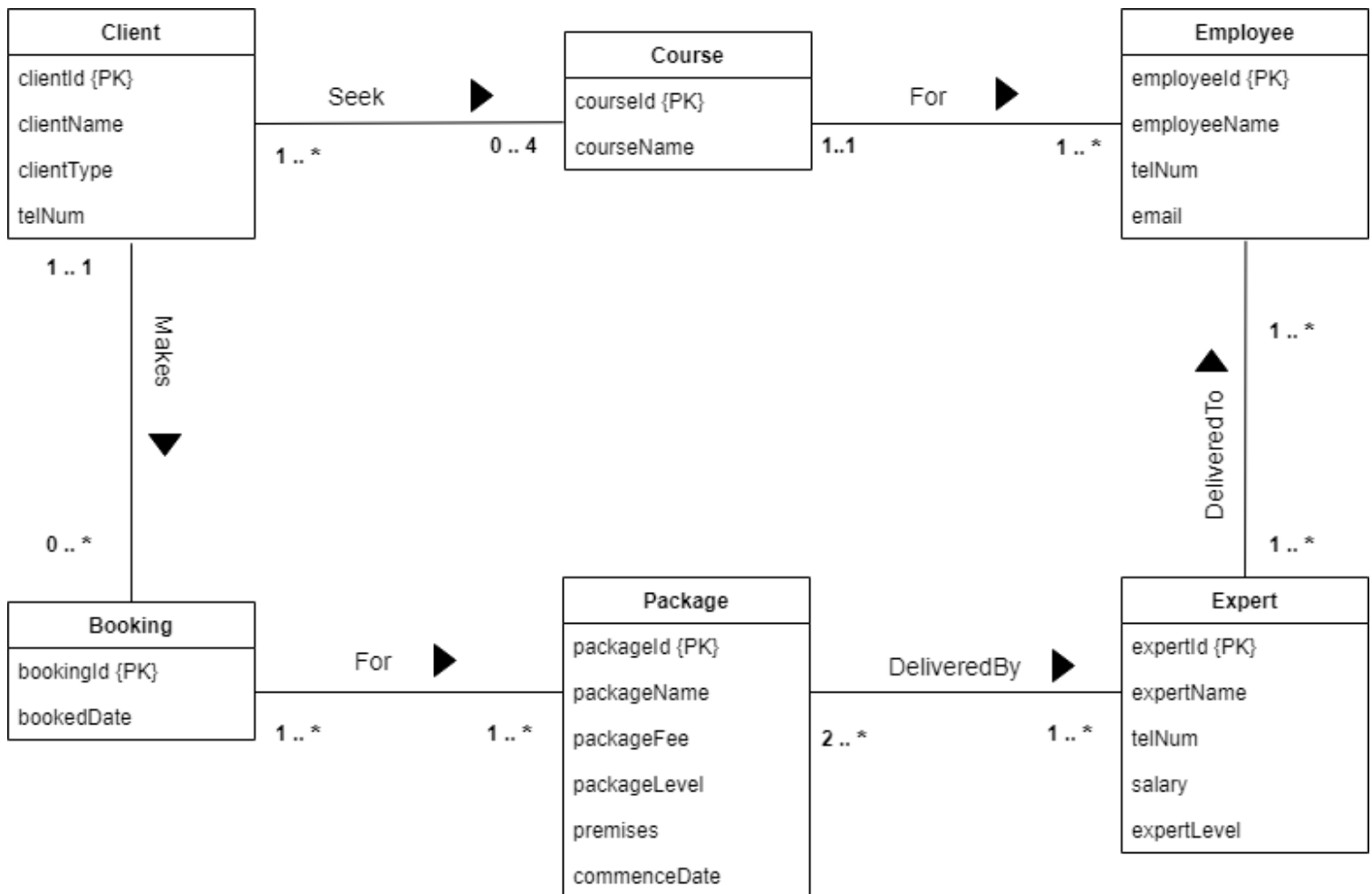
The ERD (Entity-Relationship-Diagram) is drawn in-order to develop relational databases in terms of conceptualization and implementation of functional databases. This shows a clear image view of how each table of data is inter connected.

According our case study given as an assignment, we are supposed to make a database regarding a training package booking system. The training package is booked by the client for their employees.

The client details, employee details, booking details, course details, training package details and the experts' details can be retrieved from the database which I have created.

(Anon., n.d.)

2.CONCEPTUAL DIAGRAM



(Fakhroutdinov, n.d.)

1.1. Assumptions taken when implementing the CONCEPTUAL DIAGRAM.

1.1.1. Assumptions about Entities

- I have taken six entities for the conceptual diagram. They are namely →
 - Client
 - Course
 - Employee
 - Booking
 - Package
 - Expert
- Client entity was selected because in the case study, it is mentioned that *“clients are of four types namely small companies, large companies, local government and central government departments”*.
- Course entity is selected because it is mentioned that *“clients seek for courses”*.
- Employee entity is selected because it is mentioned that *“courses are for the employees”*.
- Package is selected because it is mentioned that *“the clients book packages”*.
- Courses are the main program and the training package is the module under a course. This was an assumption regarding the separation of course and training package entities.
- Expert is selected because it is mentioned that *“STH employs a number of experts for the delivery of the training packages”*.
- Expert table is taken because my assumption about it is, employees are the people who work in the company and also follow the course. The experts are the people also employed in the company but they only deliver lessons and training packages. There by I have taken those two entities separately.
- Booking is selected as per my assumption because the client books a training package at least three months in advance. So, in-order to maintain the booking records I have used an entity named as “Booking”.
- All the entities which I have selected are binary entities. All the relationship consists of two entities. There by we can say that it is a binary relationship.

1.1.2. Assumptions about attributes

- In the client table, the clientType attribute is to identify the type of the client. It is mentioned that “*clients are small and large companies as well as local and central government departments*”. There by we can select the **type of the client**. The other attributes are to give **details** about the client.
- In the course table, the **courseName** defines the name of the course which it refers to.
- In the employee table, **employeeName** defines the name of the employee. **telNum** defines the telephone number of the employee. **email** defines the E-mail Id of the particular employee.
- Booking table was created to keep records of the booking details. This will carry the specific details about the booking. The **bookingId** and the **bookingDate**. “bookingDate” is to record the booking the date in which the booking was made. The booking date was taken to consideration because it was mentioned that a booking should be made three months before the commencement of the training package.
- Package table consists of details regarding the training package. The attributes in the entity are specific about the training package. The **packageLevel** attribute shows the levels of the packages available and the **packageFee** shows the value of each package. The **premises** attribute is to depict where the package will be delivered. The **commenceDate** attribute is added because the clients make the booking at least three months before the training package starts. Therefore, to particularly identify the date, I have taken this attribute into count.
- In the expert table, **expertName** defines the name of the expert. **telNum** defines the telephone number of the expert. **Salary** is based on the expert level of the particular expert. Therefore, **expertLevel** is taken to consideration. My assumption on the expertLevel attribute is, expert with higher expert level is able to deliver training package which are lower than it.
 - If the expertLevel is Advanced, he is able to deliver packages which are basic and intermediate too.

1.1.3. Assumptions about Primary keys

The primary key has been defined for each entity as follows. →

- Client table – clientId {PK}
 - Each and every client must have their own Client ID. So, I have taken clientId as the primary key in the client entity.
- Course – courseId {PK}
 - Each and every course must have its own Course ID. So, I have taken courseId as the primary key in the course entity.
- Employee - employeeId {PK}
 - Each and every employee must have their own Employee ID. So, I have taken employeeId as the primary key in the employee entity.
- Booking - bookingId {PK}
 - Each and every booking must have its own Booking ID. So, I have taken bookingId as the primary key in the booking entity.
- Training Package - packageId {PK}
 - Each and every package must have its own Package ID. So, I have taken packageId as the primary key in the training_package entity.
- Expert - expertId {PK}
 - Each and every expert must have their own Expert ID. So, I have taken expertId as the primary key in the expert entity

1.1.4. Assumptions about relationships

- The clients seek courses. Therefore, the relationship name between client table and course table is “Seek”.
- The client seeks course for the employee. Therefore, the relationship name between course table and employee table is “For”.
- The client makes a booking. Therefore, the relationship name between client table and booking table is “Makes”.
- A booking is made for a package. Therefore, the relationship name between booking table and package table is “IsFor”.
- A package is delivered by an expert. Therefore, the relationship name between the expert table and package table is “DeliveredBy”.
- Expert delivers the training package to the employees. Therefore, the relationship name between expert table and employee table is “DeliveredTo”.

1.1.5. Assumptions about participations and cardinality

a. Client and Course relationship

- One client may or may not seek for a course. Also, one client can seek for maximum of four courses.
- One course is sought by at least one client. One course can be sought by many clients too.

b. Course and Employee relationship

- If the course is sought, then one course is sought for at least one employee. Else, it can be sought for many employees.
- An employee should attend to only one course. Cannot attend more than one course.

c. Client and Booking relationship

- The client may or may not make a booking. If the client wants to make a booking, he can make many bookings.
- If a booking is made, that particular booking is for one and only client.

d. Booking and Package relationship

- A booking contains at least one package in minimum. A booking can have many packages in it too.
- If a booking is made, the package has to be at least in one booking. The package can be in many bookings too.

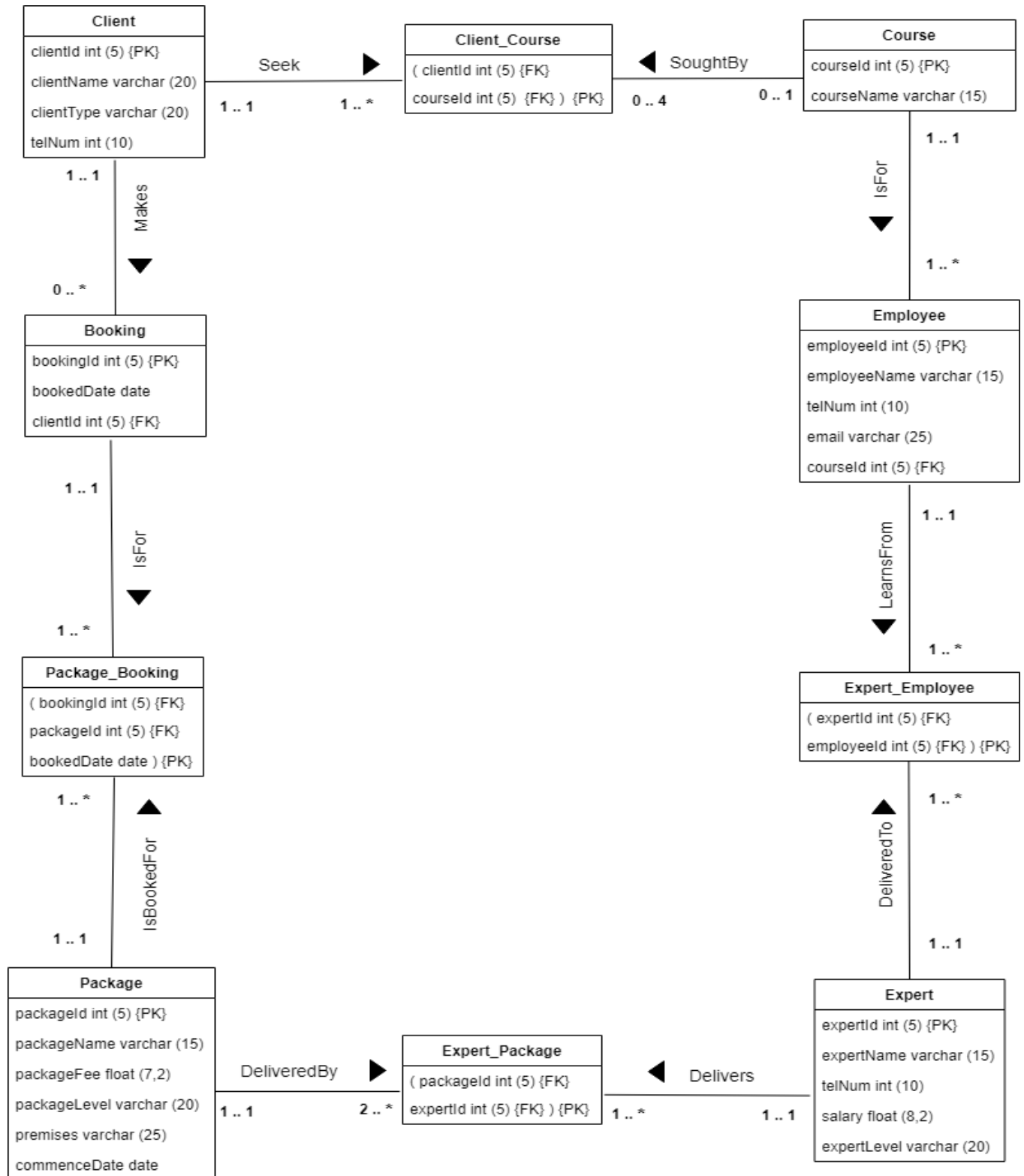
e. Package and Expert relationship

- A package should be delivered by at least one expert. A particular package can be delivered by many experts too.
- An expert should deliver minimum of at least two packages. An expert can also deliver many packages too.

f. Expert and Employee relationship

- An expert delivers the package to either one or many employees.
- An employee can receive the training package from one particular expert or else from many different experts too.

3. LOGICAL DIAGRAM



(Fakhroutdinov, n.d.)

3.1. Assumptions taken when implementing the LOGICAL DIAGRAM.

**** I have not stated the similar assumptions between LOGICAL diagram and CONCEPTUAL diagram as it would become a duplicate of data. ****

3.1.1. Assumptions about Entities

- The additional tables added into the logical diagram when compared to the conceptual diagram are →
 - Client_Course
 - Package_Booking
 - Package_Expert
 - Expert_Employee
- Client table and the course table has a many to four [* .. 4] relationship. That is considered as many to many [* .. *] eventually. Therefore, when converting that to logical diagram, an additional entity gets created. The name given to that entity is “Client_Course”.
- Package table and the booking table has a many to many [* .. *] relationship. Therefore, when converting that to logical diagram, an additional entity gets created. The name given to that entity is “Package_Booking”.
- Package table and the expert table has a many to many [* .. *] relationship. Therefore, when converting that to logical diagram, an additional entity gets created. The name given to that entity is “Expert_Package”.
- Expert table and the employee table has a many to many [* .. *] relationship. Therefore, when converting that to logical diagram, an additional entity gets created. The name given to that entity is “Expert_Employee”.
- When the course table and the employee table are considered, it has a one to many [1 .. *] relationship. Therefore, the primary key of the parent table (course table) becomes the foreign key of the child table (employee table).
- When the client table and the booking table are considered, it has a one to many [1 .. *] relationship. Therefore, the primary key of the parent table (client table) becomes the foreign key of the child table (booking table).

3.1.2. Assumptions about attributes

- **Primary keys of the client table** and the **course table** becomes a **foreign key** to the newly created **Client_Course table**. Those two foreign keys collectively become a common primary key of the Client_Course table.
- **Primary keys of the package table** and the **booking table** becomes a **foreign key** to the newly created **Package_Booking table**. Those two foreign keys collectively become a common primary key of the Package_Booking table. An additional attribute is added in this entity. By overcoming this process, the primary key turns out to be a composite primary key.
- **Primary keys of the package table** and the **expert table** becomes a **foreign key** to the newly created **Package_Expert table**. Those two foreign keys collectively become a common primary key of the Package_Expert table.
- **Primary keys of the expert table** and the **employee table** becomes a **foreign key** to the newly created **Expert_Employee table**. Those two foreign keys collectively become a common primary key of the Expert_Employee table.

The above mentioned points occur because the two entities mentioned in it are having a **many to many [* .. *] relationship**. The following which are discussed below will the attributes raised due to **one to many [1 .. *] relationship**.

- The primary key of the Client table (parent table) becomes the foreign key of the Booking table (child table).
- The primary key of the Course table (parent table) becomes the foreign key of the Employee table (child table).

3.1.3. Assumptions about participations and cardinality

- It is a must for every Client_Course table to have a record of at least one clientId but not compulsory to have a courseId. The courseId should be included only if the client seeks the course. The same clientId and courseId can be available in multiple records but only one client can seek for four courses maximum.
- It is a must for every Package_Booking table to have a record of at least one packageId and one bookingId. The same packageId and bookingId can be available in multiple records.
- It is a must for every Expert_Booking table to have a record of at least one expertId and one bookingId. The same expertId and bookingId can be available in multiple records.
- It is a must for every Expert_Employee table to have a record of at least one expertId and one employeeId. The same cexpertId and employeeId can be available in multiple records.

3.1.4. Assumptions about the data type

a. INT (*value*) →

The attributes with this data type contain only integers. This data type is given for the following attributes.

- clientId int (5) | courseId int (5) | bookingId int (5) | packageId int (5) | expertId int (5) | employeeId int (5)
→ These are for representing ID's. It is unique for every entity. Therefore, to have it as a code number I have given it as **int**.
- telNum int (10)
→ This is to record telephone numbers. A telephone number should compulsorily have 10 digits. Therefore, I have given it as **int** and maximum amount as 10.

b. FLOAT (*value, value*) →

The attributes with this data type contain only float values. That is, numbers including decimal values. For example – float (5, 2) → This represents total of 5 character including 2 decimal places. (500.25)

- packageFee float (7, 2)
→ This is to represent the cost of each package. Since the maximum value given for a package was 3000.00, I have taken 7 numbers as maximum including 2 decimal places.
- salary float (8, 2)
→ This is to represent the salary of each expert. My assumption to derive the salary was to give salary based on their expert level. An expert receives salary from multiple employees. So, in-order to keep it as a fixed amount, I have taken 8 numbers as maximum including 2 decimal places. The maximum salary I have given was 95000.00.

c. DATE →

The attribute of this data type contains only date values. The format of the date that is stored is YYYY-MM-DD.

- bookedDate DATE
→ This is to represent the date on which the booking is made.

- commenceDate DATE

➔ This is to represent the date on which the training will be started.

These two attributes are considered because a booking must be made 3 months prior to the commencement of the training package.

d. VARCHAR (value) ➔

The attribute of data type contains variable characters. This data type is given for the following attributes.

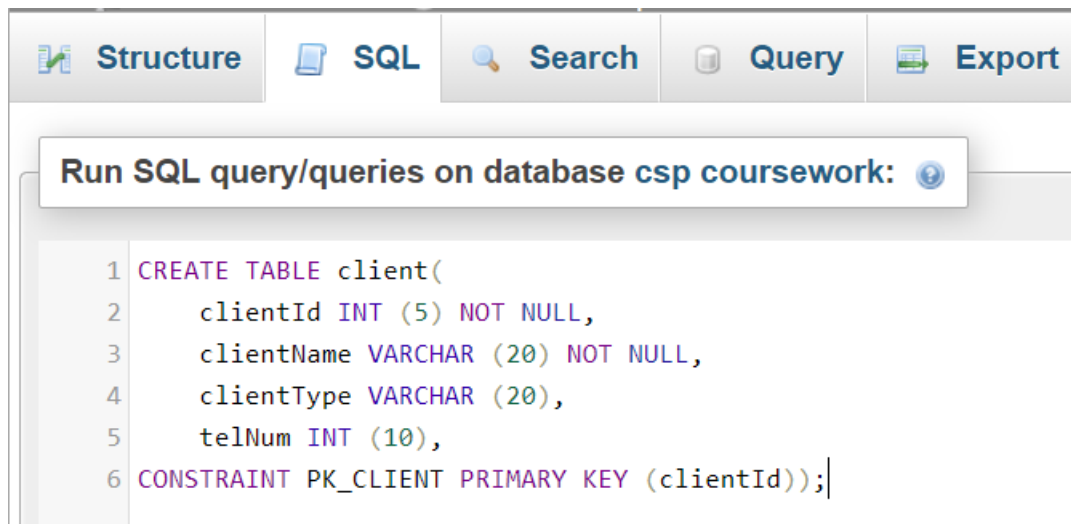
- clientName VARCHAR (20)
 - courseName VARCHAR (15)
 - packageName VARCHAR (15)
 - employeeName VARCHAR (15)
 - expertName VARCHAR (15)
- ➔ To identify the name of the entity table
- clientType VARCHAR (20) ➔ To identify the client type
- packageLevel VARCHAR (20)
 - expertLevel VARCHAR (20)
- ➔ To identify the levels
- premises VARCHAR (25) ➔ To identify the premises
- email VARCHAR (25) ➔ To identify the email address

4.CREATING TABLES

4.1. Client table

```
CREATE TABLE client (
  clientId INT (5) NOT NULL,
  clientName VARCHAR (20) NOT NULL,
  clientType VARCHAR (20),
  telNum INT (10),
  CONSTRAINT PK_CLIENT PRIMARY KEY (clientId);
```

Client table - SQL code



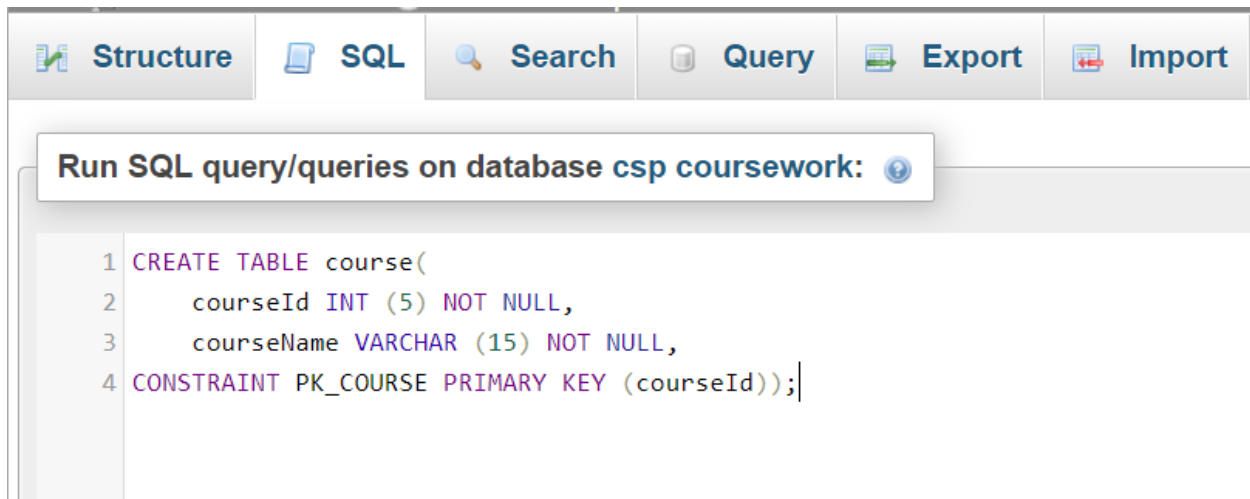
Output of the created Client table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	clientId	int(5)			No	None			Change Drop More
<input type="checkbox"/> 2	clientName	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/> 3	clientType	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/> 4	telNum	int(10)			Yes	NULL			Change Drop More

4.2. Course table

```
CREATE TABLE course (
  courseId INT (5) NOT NULL,
  courseName VARCHAR (15) NOT NULL,
  CONSTRAINT PK_COURSE PRIMARY KEY (courseId));
```

Course table – SQL code



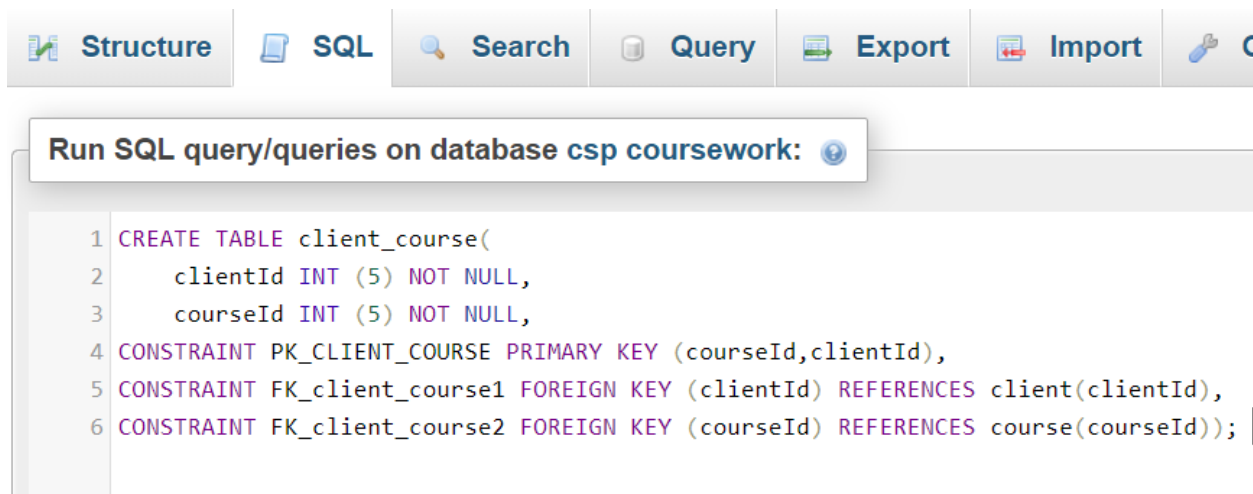
Output of the Course table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	courseId	int(5)			No	None			Change Drop More
<input type="checkbox"/> 2	courseName	varchar(15)	utf8mb4_general_ci		No	None			Change Drop More

4.3. Client_Course table

```
CREATE TABLE client_course (
  clientId INT (5) NOT NULL,
  courseId INT (5) NOT NULL,
  CONSTRAINT PK_CLIENT_COURSE PRIMARY KEY (courseId, clientId),
  CONSTRAINT FK_client_course1 FOREIGN KEY (clientId) REFERENCES client(clientId),
  CONSTRAINT FK_client_course2 FOREIGN KEY (courseId) REFERENCES course(courseId));
```

Client_Course table – SQL code



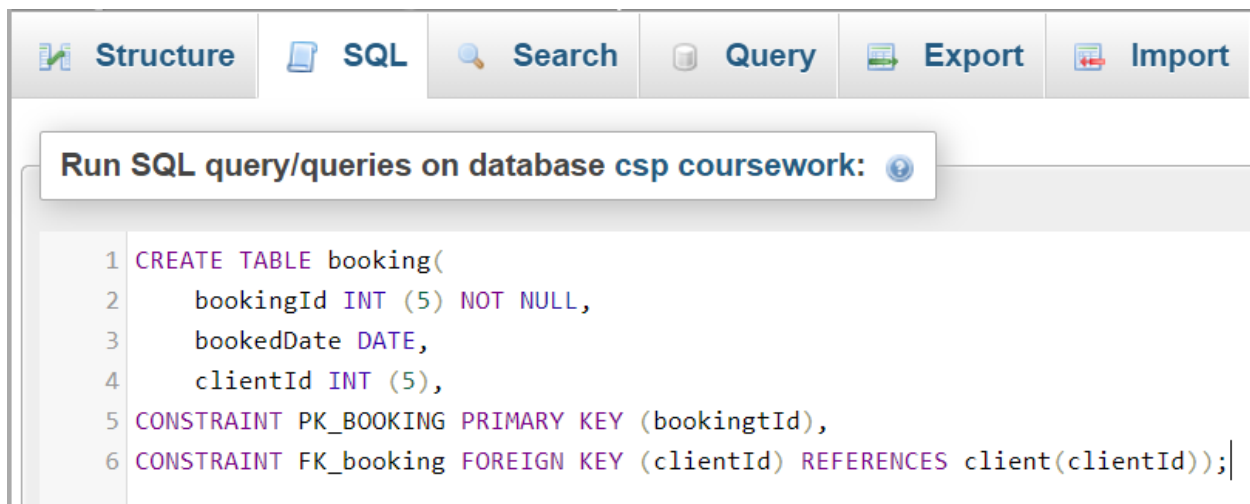
Output of the Client_Course table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	clientId	int(5)			No	None			Change Drop More
<input type="checkbox"/> 2	courseId	int(5)			No	None			Change Drop More

4.4. Booking table

```
CREATE TABLE booking (
    bookingId INT (5) NOT NULL,
    bookedDate DATE,
    clientId INT (5),
    CONSTRAINT PK_BOOKING PRIMARY KEY (bookingId),
    CONSTRAINT FK_booking FOREIGN KEY (clientId) REFERENCES client(clientId));
```

Booking table – SQL code



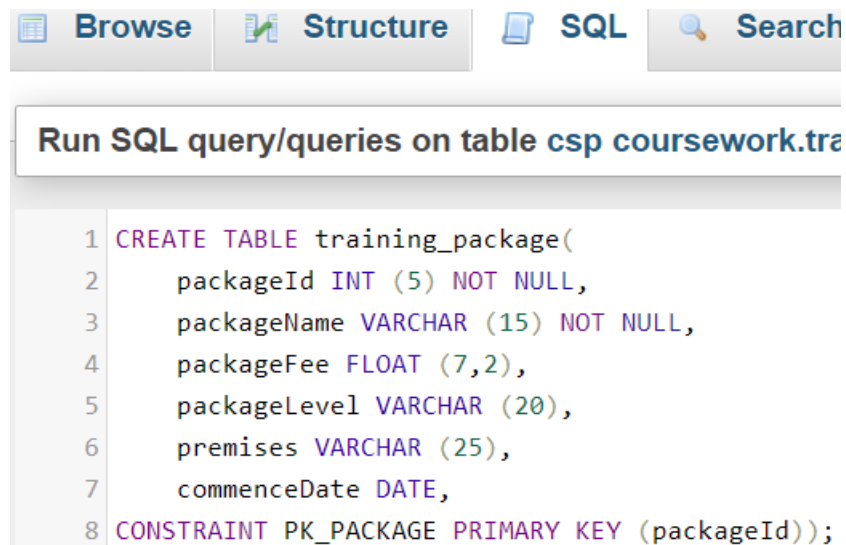
Output of the Booking table

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	bookingId	int(5)			No	None			Change Drop More
<input type="checkbox"/>	2	bookedDate	date			Yes	NULL			Change Drop More
<input type="checkbox"/>	3	clientId	int(5)			Yes	NULL			Change Drop More

4.5. Package table

```
CREATE TABLE training_package (
  packageId INT (5) NOT NULL,
  packageName VARCHAR (15) NOT NULL,
  packageFee FLOAT (7,2),
  packageLevel VARCHAR (20),
  premises VARCHAR (25),
  commenceDate DATE,
  CONSTRAINT PK_PACKAGE PRIMARY KEY (packageId));
```

Package table – SQL code



Output of the Package table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	packageId	int(5)			No	None			Change Drop More
<input type="checkbox"/> 2	packageName	varchar(15)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/> 3	packageFee	float(7,2)			Yes	NULL			Change Drop More
<input type="checkbox"/> 4	packageLevel	varchar(20)	utf8mb4_general_ci		Yes	NULL			Change Drop More
<input type="checkbox"/> 5	premises	varchar(25)	utf8mb4_general_ci		Yes	NULL			Change Drop More
<input type="checkbox"/> 6	commenceDate	date			Yes	NULL			Change Drop More

4.6. Package_Booking table

```
CREATE TABLE package_booking (
    packageId INT (5) NOT NULL,
    bookingId INT (5) NOT NULL,
    bookedDate DATE,
    CONSTRAINT PK_PACKAGE_BOOKING PRIMARY KEY (packageId, bookingId, bookedDate),
    CONSTRAINT FK_package_booking1 FOREIGN KEY (packageId) REFERENCES
training_package(packageId),
    CONSTRAINT FK_package_booking2 FOREIGN KEY (bookingId) REFERENCES
booking(bookingId));
```

Package_Booking table – SQL code

The screenshot shows a database management tool with a top navigation bar containing icons and labels for Structure, SQL, Search, Query, Export, Import, and Operations. Below this is a banner that reads "Run SQL query/queries on database csp coursework:". The main area displays the following SQL code:

```
1 CREATE TABLE package_booking(
2     packageId INT (5) NOT NULL,
3     bookingId INT (5) NOT NULL,
4     bookedDate DATE,
5     CONSTRAINT PK_PACKAGE_BOOKING PRIMARY KEY (packageId, bookingId, bookedDate),
6     CONSTRAINT FK_package_booking1 FOREIGN KEY (packageId) REFERENCES training_package(packageId),
7     CONSTRAINT FK_package_booking2 FOREIGN KEY (bookingId) REFERENCES booking(bookingId));
```

Output of the Package_Booking table

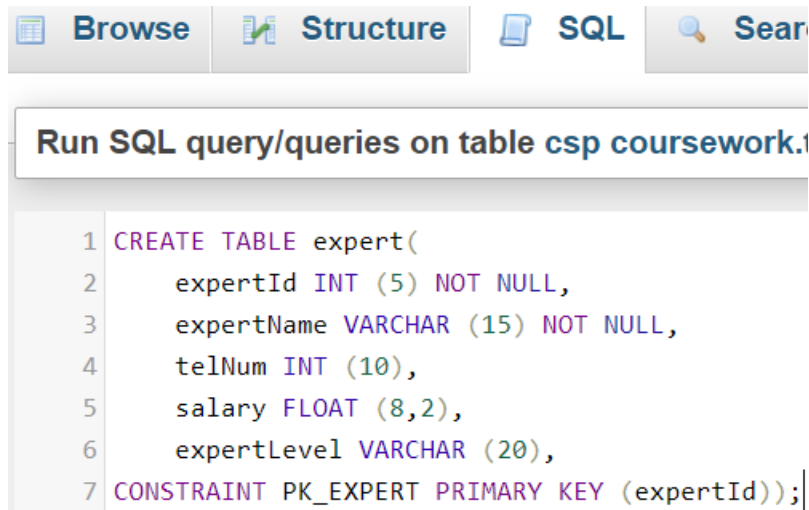
The screenshot shows the same database management tool interface, but with the "Table structure" tab selected. It displays the structure of the package_booking table with the following columns:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	packageId 🔑	int(5)			No	None			Change Drop More
<input type="checkbox"/> 2	bookingId 🔑	int(5)			No	None			Change Drop More
<input type="checkbox"/> 3	bookedDate 🔑	date			No	None			Change Drop More

4.7. Expert table

```
CREATE TABLE expert (
    expertId INT (5) NOT NULL,
    expertName VARCHAR (15) NOT NULL,
    telNum INT (10),
    salary FLOAT (8,2),
    expertLevel VARCHAR (20),
    CONSTRAINT PK_EXPERT PRIMARY KEY (expertId));
```

Expert table – SQL code



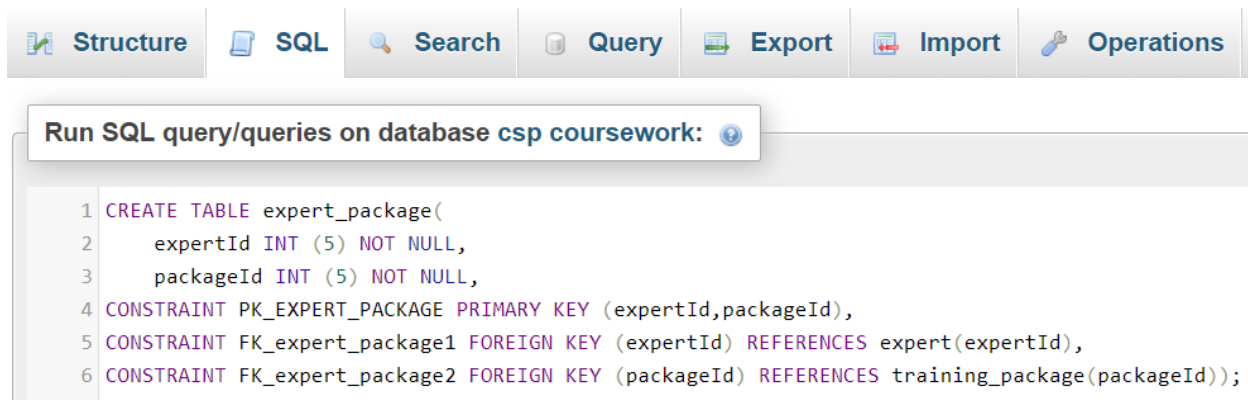
Output of the Expert table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	expertId	int(5)			No	None			Change Drop More
<input type="checkbox"/> 2	expertName	varchar(15)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/> 3	telNum	int(10)			Yes	NULL			Change Drop More
<input type="checkbox"/> 4	salary	float(8,2)			Yes	NULL			Change Drop More
<input type="checkbox"/> 5	expertLevel	varchar(20)	utf8mb4_general_ci		Yes	NULL			Change Drop More

4.8. Expert_Package table

```
CREATE TABLE expert_package (
    expertId INT (5) NOT NULL,
    packageId INT (5) NOT NULL,
    CONSTRAINT PK_EXPERT_PACKAGE PRIMARY KEY (expertId, packageId),
    CONSTRAINT FK_expert_package1 FOREIGN KEY (expertId) REFERENCES expert(expertId),
    CONSTRAINT FK_expert_package2 FOREIGN KEY (packageId) REFERENCES
training_package(packageId));
```

Expert_Package table – SQL code



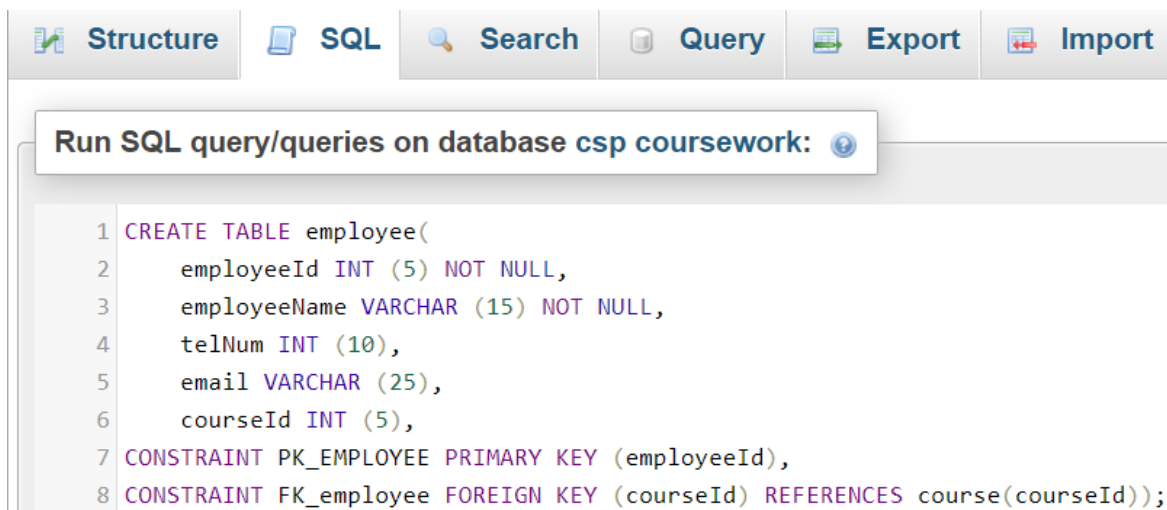
Output of the Expert_Package table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	expertId 🔑	int(5)			No	None			Change Drop More
<input type="checkbox"/> 2	packageId 🔑	int(5)			No	None			Change Drop More

4.9. Employee table

```
CREATE TABLE employee (
    employeeId INT (5) NOT NULL,
    employeeName VARCHAR (15) NOT NULL,
    telNum INT (10),
    email VARCHAR (25),
    courseId INT (5),
    CONSTRAINT PK_EMPLOYEE PRIMARY KEY (employeeId),
    CONSTRAINT FK_employee FOREIGN KEY (courseId) REFERENCES course(courseId));
```

Employee table – SQL code



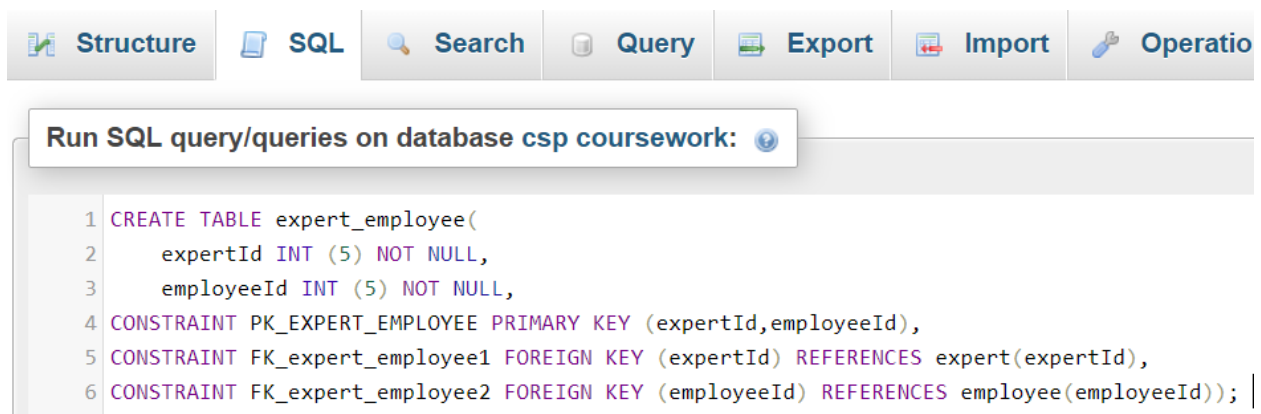
Output of the Employee table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	employeeId	int(5)			No	None			Change Drop More
<input type="checkbox"/> 2	employeeName	varchar(15)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/> 3	telNum	int(10)			Yes	NULL			Change Drop More
<input type="checkbox"/> 4	email	varchar(25)	utf8mb4_general_ci		Yes	NULL			Change Drop More
<input type="checkbox"/> 5	courseId	int(5)			Yes	NULL			Change Drop More

4.10. Expert_Employee table

```
CREATE TABLE expert_employee (
    expertId INT (5) NOT NULL,
    employeeId INT (5) NOT NULL,
    CONSTRAINT PK_EXPERT_EMPLOYEE PRIMARY KEY (expertId, employeeId),
    CONSTRAINT FK_expert_employee1 FOREIGN KEY (expertId) REFERENCES expert(expertId),
    CONSTRAINT FK_expert_employee2 FOREIGN KEY (employeeId) REFERENCES
employee(employeeId));
```

Expert_Employee table – SQL code



Output of the Expert_Employee table

The screenshot shows a database management tool interface with a toolbar at the top containing buttons for Browse, Structure, SQL, Search, Insert, Export, Import, and Privileges. Below the toolbar, there are two tabs: "Table structure" (selected) and "Relation view". The main area displays the table structure for the Expert_Employee table.

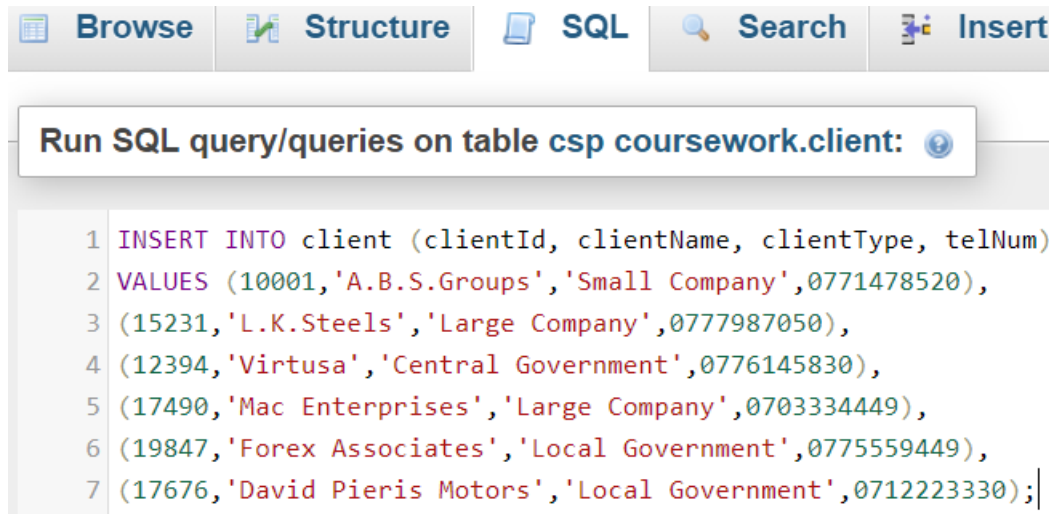
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	expertId 🔑	int(5)			No	None			🔧 Change 🚫 Drop ▼ More
<input type="checkbox"/> 2	employeeId 🔑	int(5)			No	None			🔧 Change 🚫 Drop ▼ More

5.INSERTING TABLES

5.1. Client Table

```
INSERT INTO client (clientId, clientName, clientType, telNum)
VALUES (10001,'A.B.S.Groups','Small Company',0771478520),
(15231,'L.K.Steels','Large Company',0777987050),
(12394,'Virtusa','Central Government',0776145830),
(17490,'Mac Enterprises','Large Company',0703334449),
(19847,'Forex Associates','Local Government',0775559449),
(17676,'David Pieris Motors','Local Government',0712223330);
```

Client table – Insert SQL code



Output of the inserted Client table

		clientId	clientName	clientType	telNum
<input type="checkbox"/>	Edit Copy Delete	10001	A.B.S.Groups	Small Company	771478520
<input type="checkbox"/>	Edit Copy Delete	12394	Virtusa	Central Government	776145830
<input type="checkbox"/>	Edit Copy Delete	15231	L.K.Steels	Large Company	777987050
<input type="checkbox"/>	Edit Copy Delete	17490	Mac Enterprises	Large Company	703334449
<input type="checkbox"/>	Edit Copy Delete	17676	David Pieris Motors	Local Government	712223330
<input type="checkbox"/>	Edit Copy Delete	19847	Forex Associates	Local Government	775559449

5.2. Course table

```
INSERT INTO course (courseId, courseName)
VALUES (20230,'Web Designing'),
(20231,'Database'),
(20232,'Image Creations'),
(20233,'Video Creations')
```

Course table – Insert SQL code



Output of the inserted Course table

				courseId	courseName
<input type="checkbox"/>	Edit	Copy	Delete	20230	Web Designing
<input type="checkbox"/>	Edit	Copy	Delete	20231	Database
<input type="checkbox"/>	Edit	Copy	Delete	20232	Image Creations
<input type="checkbox"/>	Edit	Copy	Delete	20233	Video Creations

























5.3. Client_Course table

```
INSERT INTO client_course (clientId, courseId)
VALUES (10001,20230),
(10001,20232),
(15231,20233),
(12394,20233),
(17490,20231),
(19847,20230),
(19847,20232),
(17676,20231)
```

Client_Course – Insert SQL code

```
1 INSERT INTO client_course (clientId, courseId)
2 VALUES (10001,20230),
3 (10001,20232),
4 (15231,20233),
5 (12394,20233),
6 (17490,20231),
7 (19847,20230),
8 (19847,20232),
9 (17676,20231)
```

Output of the inserted Client_Course table

<div>←T→</div>				clientId	courseId
<input type="checkbox"/>	 Edit	 Copy	 Delete	10001	20230
<input type="checkbox"/>	 Edit	 Copy	 Delete	19847	20230
<input type="checkbox"/>	 Edit	 Copy	 Delete	17490	20231
<input type="checkbox"/>	 Edit	 Copy	 Delete	17676	20231
<input type="checkbox"/>	 Edit	 Copy	 Delete	10001	20232
<input type="checkbox"/>	 Edit	 Copy	 Delete	19847	20232
<input type="checkbox"/>	 Edit	 Copy	 Delete	12394	20233
<input type="checkbox"/>	 Edit	 Copy	 Delete	15231	20233






















5.4. Booking table

```
INSERT INTO booking (bookingId, bookedDate, clientId)
VALUES (45650,'2020-03-01',10001),
(45660,'2020-02-17',15231),
(45670,'2020-02-20',12394),
(45680,'2020-03-05',17676),
(45690,'2020-03-10',19847),
(45700,'2020-02-28',17490),
(45710,'2020-03-15',15231)
```

Booking table – Insert SQL code

```
1 INSERT INTO booking (bookingId, bookedDate, clientId)
2 VALUES (45650, '2020-03-01', 10001),
3 (45660, '2020-02-17', 15231),
4 (45670, '2020-02-20', 12394),
5 (45680, '2020-03-05', 17676),
6 (45690, '2020-03-10', 19847),
7 (45700, '2020-02-28', 17490),
8 (45710, '2020-03-15', 15231)|
```

Output of the inserted Booking table

← T →				bookingId	bookedDate	clientId
<input type="checkbox"/>		Edit		Copy		Delete
				45650	2020-03-01	10001
<input type="checkbox"/>		Edit		Copy		Delete
				45660	2020-02-17	15231
<input type="checkbox"/>		Edit		Copy		Delete
				45670	2020-02-20	12394
<input type="checkbox"/>		Edit		Copy		Delete
				45680	2020-03-05	17676
<input type="checkbox"/>		Edit		Copy		Delete
				45690	2020-03-10	19847
<input type="checkbox"/>		Edit		Copy		Delete
				45700	2020-02-28	17490
<input type="checkbox"/>		Edit		Copy		Delete
				45710	2020-03-15	15231

5.5. Package table

INSERT INTO training_package (packageId, packageName, packageFee, packageLevel, premises, commenceDate)

VALUES (52105,'Adobe',200.00,'Basic','STH IT Labs','2020-07-05'),
 (52110,'Adobe',500.00,'Intermediate','Client premises','2020-07-07'),
 (52115,'Adobe',1000.00,'Advanced','STH IT Labs','2020-07-12'),
 (52120,'Adobe',2000.00,'Customised','Client premises','2020-07-14'),
 (52125,'WorkBench',200.00,'Basic','STH IT Labs','2020-07-05'),
 (52130,'WorkBench',500.00,'Intermediate','Client premises','2020-07-07'),
 (52135,'WorkBench',1000.00,'Advanced','STH IT Labs','2020-07-12'),
 (52140,'WorkBench',2500.00,'Customised','Client premises','2020-07-14')

Package table – Insert SQL code

```
1 INSERT INTO training_package (packageId, packageName, packageFee, packageLevel, premises,
  commenceDate)
2 VALUES (52105,'Adobe',200.00,'Basic','STH IT Labs','2020-07-05'),
3 (52110,'Adobe',500.00,'Intermediate','Client premises','2020-07-07'),
4 (52115,'Adobe',1000.00,'Advanced','STH IT Labs','2020-07-12'),
5 (52120,'Adobe',2000.00,'Customised','Client premises','2020-07-14'),
6 (52125,'WorkBench',200.00,'Basic','STH IT Labs','2020-07-05'),
7 (52130,'WorkBench',500.00,'Intermediate','Client premises','2020-07-07'),
8 (52135,'WorkBench',1000.00,'Advanced','STH IT Labs','2020-07-12'),
9 (52140,'WorkBench',2500.00,'Customised','Client premises','2020-07-14')
```

Output of the inserted Package table

		packageId	packageName	packageFee	packageLevel	premises	commenceDate
<input type="checkbox"/>	Edit Copy Delete	52140	WorkBench	2500.00	Customised	Client premises	2020-07-14
<input type="checkbox"/>	Edit Copy Delete	52135	WorkBench	1000.00	Advanced	STH IT Labs	2020-07-12
<input type="checkbox"/>	Edit Copy Delete	52130	WorkBench	500.00	Intermediate	Client premises	2020-07-07
<input type="checkbox"/>	Edit Copy Delete	52125	WorkBench	200.00	Basic	STH IT Labs	2020-07-05
<input type="checkbox"/>	Edit Copy Delete	52120	Adobe	2000.00	Customised	Client premises	2020-07-14
<input type="checkbox"/>	Edit Copy Delete	52115	Adobe	1000.00	Advanced	STH IT Labs	2020-07-12
<input type="checkbox"/>	Edit Copy Delete	52110	Adobe	500.00	Intermediate	Client premises	2020-07-07
<input type="checkbox"/>	Edit Copy Delete	52105	Adobe	200.00	Basic	STH IT Labs	2020-07-05

























5.6. Package_Booking table

```
INSERT INTO package_booking (packageId, bookingId, bookedDate)
VALUES (52140,45650,'2020-03-01'),
(52140,45660,'2020-02-17'),
(52125,45670,'2020-02-20'),
(52125,45650,'2020-03-01'),
(52110,45690,'2020-03-10'),
(52110,45700,'2020-02-28'),
(52120,45710,'2020-03-15'),
(52120,45690,'2020-03-15')
```

Package_Booking table – Insert SQL code

```
1 INSERT INTO package_booking (packageId, bookingId, bookedDate)
2 VALUES (52140,45650,'2020-03-01'),
3 (52140,45660,'2020-02-17'),
4 (52125,45670,'2020-02-20'),
5 (52125,45650,'2020-03-01'),
6 (52110,45690,'2020-03-10'),
7 (52110,45700,'2020-02-28'),
8 (52120,45710,'2020-03-15'),
9 (52120,45690,'2020-03-15')
```

Output of the inserted Package_Booking table

← T →				packageId	bookingId	bookedDate
<input type="checkbox"/>		Edit	 Copy  Delete	52110	45690	2020-03-10
<input type="checkbox"/>		Edit	 Copy  Delete	52110	45700	2020-02-28
<input type="checkbox"/>		Edit	 Copy  Delete	52120	45690	2020-03-15
<input type="checkbox"/>		Edit	 Copy  Delete	52120	45710	2020-03-15
<input type="checkbox"/>		Edit	 Copy  Delete	52125	45650	2020-03-01
<input type="checkbox"/>		Edit	 Copy  Delete	52125	45670	2020-02-20
<input type="checkbox"/>		Edit	 Copy  Delete	52140	45650	2020-03-01
<input type="checkbox"/>		Edit	 Copy  Delete	52140	45660	2020-02-17

5.7. Expert table

```
INSERT INTO expert (expertId, expertName, telNum, salary, expertLevel)
VALUES (79000,'John',0767474764,50000.00,'Intermediate'),
(79005,'Peter',0762323456,52000.00,'Intermediate'),
(79010,'Micheal',0774858159,75000.00,'Advanced'),
(79015,'James',0717878978,90000.00,'Customised'),
(79020,'Mary',0753636536,95000.00,'Customised'),
(79025,'Alexa',0778956231,95000.00,'Customised')
```

Expert table – Insert SQL code

```
1 INSERT INTO expert (expertId, expertName, telNum, salary, expertLevel)
2 VALUES (79000,'John',0767474764,50000.00,'Intermediate'),
3 (79005,'Peter',0762323456,52000.00,'Intermediate'),
4 (79010,'Micheal',0774858159,75000.00,'Advanced'),
5 (79015,'James',0717878978,90000.00,'Customised'),
6 (79020,'Mary',0753636536,95000.00,'Customised'),
7 (79025,'Alexa',0778956231,95000.00,'Customised')|
```

Output of the inserted Expert table

		expertId	expertName	telNum	salary	expertLevel
<input type="checkbox"/>	Edit Copy Delete	79000	John	767474764	50000.00	Intermediate
<input type="checkbox"/>	Edit Copy Delete	79005	Peter	762323456	52000.00	Intermediate
<input type="checkbox"/>	Edit Copy Delete	79010	Micheal	774858159	75000.00	Advanced
<input type="checkbox"/>	Edit Copy Delete	79015	James	717878978	90000.00	Customised
<input type="checkbox"/>	Edit Copy Delete	79020	Mary	753636536	95000.00	Customised
<input type="checkbox"/>	Edit Copy Delete	79025	Alexa	778956231	95000.00	Customised
























5.8. Expert_Package table

```
INSERT INTO expert_package (expertId, packageId)
VALUES (79000,52130),
(79000,52125),
(79005,52130),
(79005,52125),
(79010,52115),
(79010,52110),
(79010,52105),
(79020,52120),
(79020,52140)
```

Expert_Package table – Insert SQL code

```
1 INSERT INTO expert_package (expertId,packageId)
2 VALUES (79000,52130),
3 (79000,52125),
4 (79005,52130),
5 (79005,52125),
6 (79010,52115),
7 (79010,52110),
8 (79010,52105),
9 (79020,52120),
10 (79020,52140)|
```

Output of the inserted Expert_Package table

←T→						expertId	packageId	
<input type="checkbox"/>		Edit		Copy		Delete	79000	52125
<input type="checkbox"/>		Edit		Copy		Delete	79000	52130
<input type="checkbox"/>		Edit		Copy		Delete	79005	52125
<input type="checkbox"/>		Edit		Copy		Delete	79005	52130
<input type="checkbox"/>		Edit		Copy		Delete	79010	52105
<input type="checkbox"/>		Edit		Copy		Delete	79010	52110
<input type="checkbox"/>		Edit		Copy		Delete	79010	52115
<input type="checkbox"/>		Edit		Copy		Delete	79020	52120
<input type="checkbox"/>		Edit		Copy		Delete	79020	52140

5.9. Employee table

```
INSERT INTO employee (employeeId, employeeName, telNum, email, courseId)
VALUES (90510, 'Vijay', 0779459456, 'vijay@gmail.com', 20231),
(90515, 'Sri Ram', 0765454122, 'sr.ram@hotmail.com', 20233),
(90520, 'Robert', 0779412321, 'robert@gmail.com', 20230),
(90525, 'Ragu', 0755512256, 'ragu@yahoo.com', 20232),
(90530, 'Pooja', 0779459456, 'vijay@gmail.com', 20233),
(90535, 'Angelina', 0766333654, 'angelina@hotmail.com', 20230),
(90540, 'Percy', 0723311166, 'percy@gmail.com', 20232)
```

Employee table – Insert SQL code

```
1 INSERT INTO employee (employeeId, employeeName, telNum, email, courseId)
2 VALUES (90510, 'Vijay', 0779459456, 'vijay@gmail.com', 20231),
3 (90515, 'Sri Ram', 0765454122, 'sr.ram@hotmail.com', 20233),
4 (90520, 'Robert', 0779412321, 'robert@gmail.com', 20230),
5 (90525, 'Ragu', 0755512256, 'ragu@yahoo.com', 20232),
6 (90530, 'Pooja', 0779459456, 'vijay@gmail.com', 20233),
7 (90535, 'Angelina', 0766333654, 'angelina@hotmail.com', 20230),
8 (90540, 'Percy', 0723311166, 'percy@gmail.com', 20232)|
```

Output of the inserted Employee table

	employeeId	employeeName	telNum	email	courseId
<input type="checkbox"/> Edit Copy Delete	90510	Vijay	779459456	vijay@gmail.com	20231
<input type="checkbox"/> Edit Copy Delete	90515	Sri Ram	765454122	sr.ram@hotmail.com	20233
<input type="checkbox"/> Edit Copy Delete	90520	Robert	779412321	robert@gmail.com	20230
<input type="checkbox"/> Edit Copy Delete	90525	Ragu	755512256	ragu@yahoo.com	20232
<input type="checkbox"/> Edit Copy Delete	90530	Pooja	779459456	vijay@gmail.com	20233
<input type="checkbox"/> Edit Copy Delete	90535	Angelina	766333654	angelina@hotmail.com	20230
<input type="checkbox"/> Edit Copy Delete	90540	Percy	723311166	percy@gmail.com	20232































5.10. Expert_Employee table

```
INSERT INTO expert_employee (expertId, employeeId)
VALUES (79000,90510),
(79000,90515),
(79000,90540),
(79005,90520),
(79005,90525),
(79005,90530),
(79010,90535),
(79010,90540),
(79020,90540),
(79020,90510)
```

Expert_Employee table – Insert SQL code

```
1 INSERT INTO expert_employee (expertId,employeeId)
2 VALUES (79000,90510),
3 (79000,90515),
4 (79000,90540),
5 (79005,90520),
6 (79005,90525),
7 (79005,90530),
8 (79010,90535),
9 (79010,90540),
10 (79020,90540),
11 (79020,90510)|
```

Output of the inserted Expert_Employee table

←T→						expertId	employeeId	
<input type="checkbox"/>		Edit		Copy		Delete	79000	90510
<input type="checkbox"/>		Edit		Copy		Delete	79000	90515
<input type="checkbox"/>		Edit		Copy		Delete	79000	90540
<input type="checkbox"/>		Edit		Copy		Delete	79005	90520
<input type="checkbox"/>		Edit		Copy		Delete	79005	90525
<input type="checkbox"/>		Edit		Copy		Delete	79005	90530
<input type="checkbox"/>		Edit		Copy		Delete	79010	90535
<input type="checkbox"/>		Edit		Copy		Delete	79010	90540
<input type="checkbox"/>		Edit		Copy		Delete	79020	90510
<input type="checkbox"/>		Edit		Copy		Delete	79020	90540

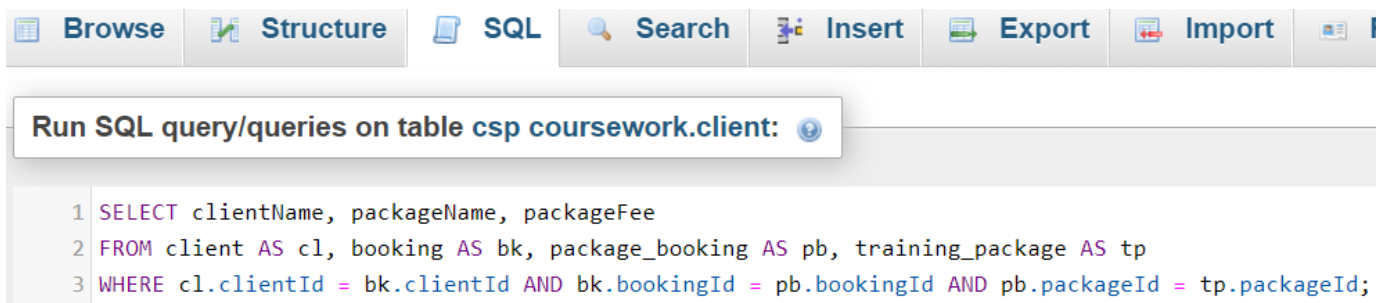
6.Select Queries

6.1. Query 1

Display the package name and the package fee of which the client has booked

```
SELECT clientName, packageName, packageFee
FROM client AS cl, booking AS bk, package_booking AS pb, training_package AS tp
WHERE cl.clientId = bk.clientId AND bk.bookingId = pb.bookingId AND pb.packageId =
tp.packageId;
```

Select query 1 – code



Select query 1 – output

clientName	packageName	packageFee
A.B.S.Groups	WorkBench	200.00
A.B.S.Groups	WorkBench	2500.00
Virtusa	WorkBench	200.00
L.K.Steels	WorkBench	2500.00
L.K.Steels	Adobe	2000.00
Mac Enterprises	Adobe	500.00
Forex Associates	Adobe	500.00
Forex Associates	Adobe	2000.00

Tables involved

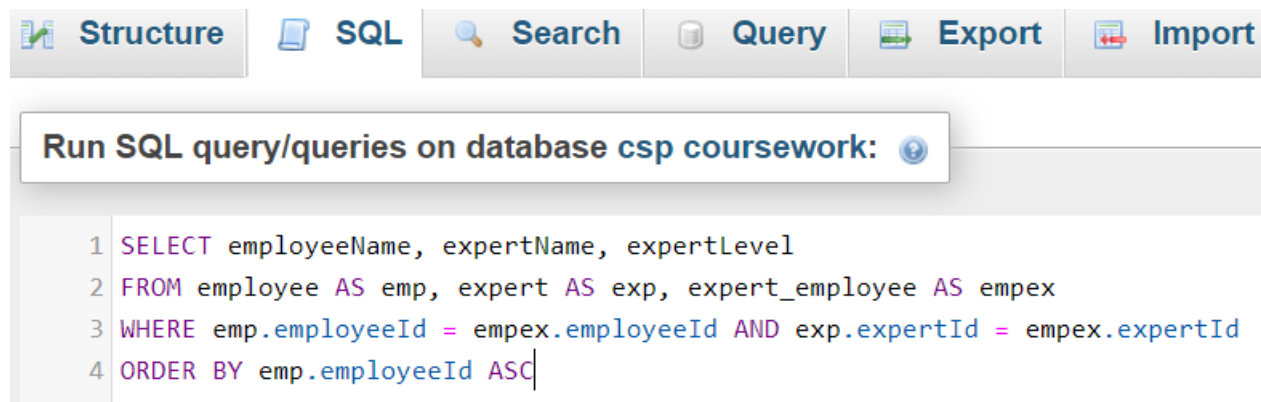
- Client table
- Booking table
- Package_Booking table
- Training_Package table

6.2. Query 2

Display expert name and expert level of a particular employee

```
SELECT employeeName, expertName, expertLevel
FROM employee AS emp, expert AS exp, expert_employee AS empex
WHERE emp.employeeId = empex.employeeId AND exp.expertId = empex.expertId
```

Select query 2 – code



Select query 2 – output

employeeName	expertName	expertLevel
Vijay	Mary	Customised
Vijay	John	Intermediate
Sri Ram	John	Intermediate
Robert	Peter	Intermediate
Ragu	Peter	Intermediate
Pooja	Peter	Intermediate
Angelina	Micheal	Advanced
Percy	Mary	Customised
Percy	John	Intermediate
Percy	Micheal	Advanced

Tables involved

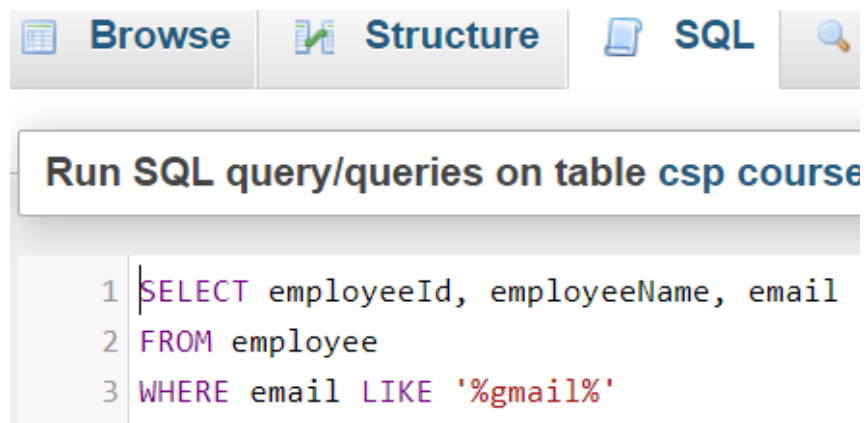
- Employee table
- Expert table
- Expert_Employee table

6.3. Query 3

Select employees who use Gmail

```
SELECT employeeId, employeeName, email  
FROM employee  
WHERE email LIKE '%gmail%'
```

Select query 3 – code



Select query 3 – output

employeeId	employeeName	email
90510	Vijay	vijay@gmail.com
90520	Robert	robert@gmail.com
90530	Pooja	vijay@gmail.com
90540	Percy	percy@gmail.com

Tables involved

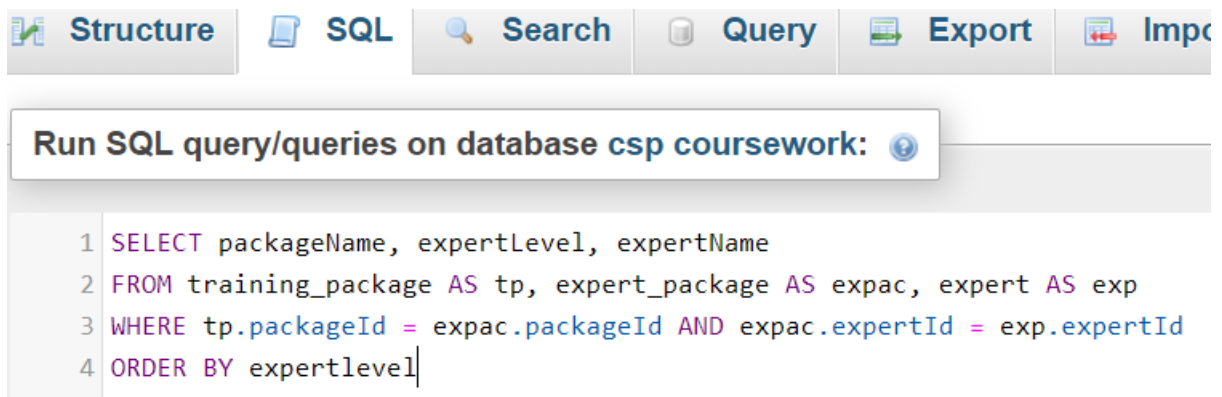
- Employee table

6.4. Query 4

Select experts according to their expert level in training package

```
SELECT packageName, expertLevel, expertName
FROM training_package AS tp, expert_package AS expac, expert AS exp
WHERE tp.packageId = expac.packageId AND expac.expertId = exp.expertId
ORDER BY level
```

Select query 4 – code



Select query 4 – output

packageName	expertLevel	expertName
Adobe	Advanced	Micheal
Adobe	Advanced	Micheal
Adobe	Advanced	Micheal
Adobe	Customised	Mary
WorkBench	Customised	Mary
WorkBench	Intermediate	John
WorkBench	Intermediate	John
WorkBench	Intermediate	Peter
WorkBench	Intermediate	Peter

Tables involved

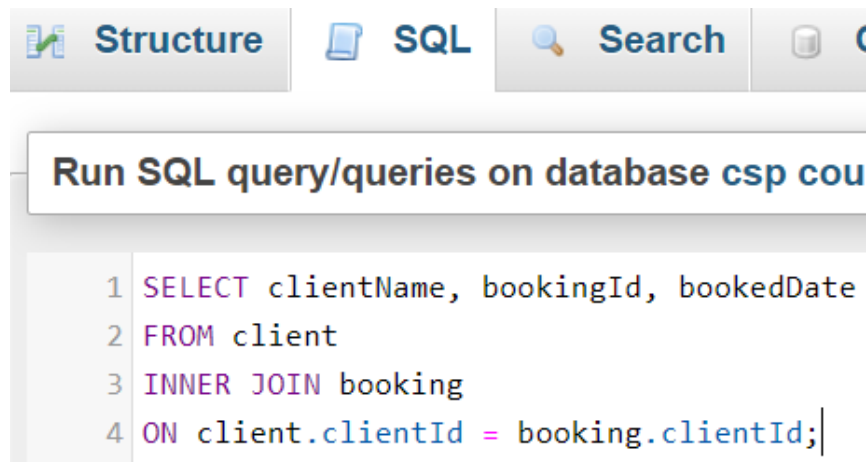
- Training_Package table
- Expert table
- Expert_Package table

6.5. Query 5

Display the booking ID and the booked date of which the customer booked (Anon., n.d.)

```
SELECT clientName, bookingId, bookedDate
FROM client
INNER JOIN booking
ON client.clientId = booking.clientId;
```

Select query 5 – code



Select query 5 – output

clientName	bookingId	bookedDate
A.B.S.Groups	45650	2020-03-01
L.K.Steels	45660	2020-02-17
Virtusa	45670	2020-02-20
David Pieris Motors	45680	2020-03-05
Forex Associates	45690	2020-03-10
Mac Enterprises	45700	2020-02-28
L.K.Steels	45710	2020-03-15

Tables involved

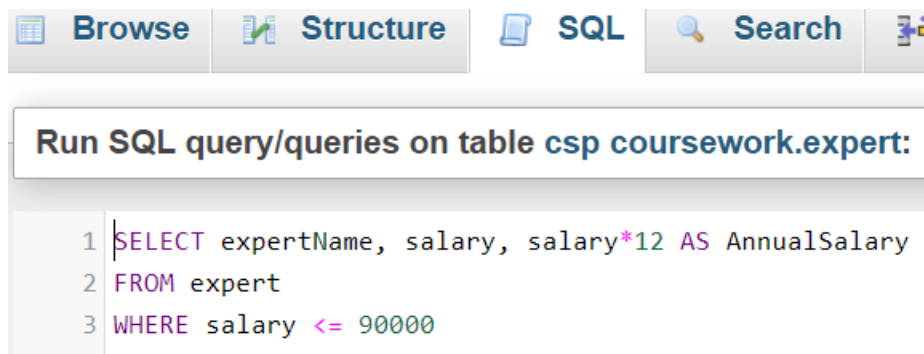
- Client table

6.6. Query 6

Select experts who receive salary less or equal to 90,000 and calculate their annual salary

```
SELECT expertName, salary, salary*12 AS AnnualSalary
FROM expert
WHERE salary <= 90000
```

Select query 6 – code



Select query 6 – output

expertName	salary	AnnualSalary
John	50000.00	600000.00
Peter	52000.00	624000.00
Micheal	75000.00	900000.00
James	90000.00	1080000.00

Tables involved

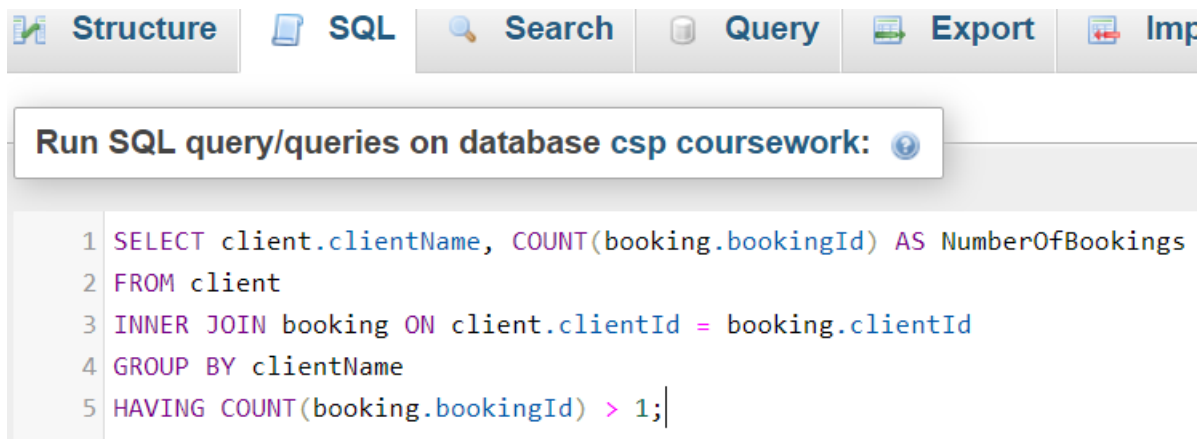
- Expert table

6.7. Query 7

Select clients who has made more than one booking and display the number of bookings made

```
SELECT client.clientName, COUNT(booking.bookingId) AS NumberOfBookings
FROM client
INNER JOIN booking ON client.clientId = booking.clientId
GROUP BY clientName
HAVING COUNT (booking.bookingId) > 1;
```

Select query 7 – code



Select query 5 – output

clientName	NumberOfBookings
L.K.Steels	2

Tables involved

- Client table
- Booking table

6.8. Query 8

Display the employees and their respective courses they do with their telephone number

```
SELECT employee.employeeName, course.courseName, employee.telNum
FROM employee
INNER JOIN course ON course.courseId = employee.courseId
GROUP BY employeeName;
```

Select query 8 – code



Select query 5 – output

employeeName	courseName	telNum
Angelina	Web Designing	766333654
Percy	Image Creations	723311166
Pooja	Video Creations	779459456
Ragu	Image Creations	755512256
Robert	Web Designing	779412321
Sri Ram	Video Creations	765454122
Vijay	Database	779459456

Tables involved

- Employee table
- Course table

7. References

- Anon., n.d. *Learning.westminster.ac.uk*. [Online]
Available at: https://learning.westminster.ac.uk/ultra/courses/_78654_1/cl/outline
[Accessed 10 06 2020].
- Anon., n.d. *SQL INNER JOIN Keyword*. [Online]
Available at: https://www.w3schools.com/sql/sql_join_inner.asp
[Accessed 25 06 2020].
- Fakhroutdinov, K., n.d. *UML Multiplicity and Collections - defining and using multiplicity and collections in UML - lower and upper bounds, cardinality, order, unique..* [Online]
Available at: <https://www.uml-diagrams.org/multiplicity.html#:~:text=Multiplicity%20in%20UML%20allows>
[Accessed 20 06 2020].