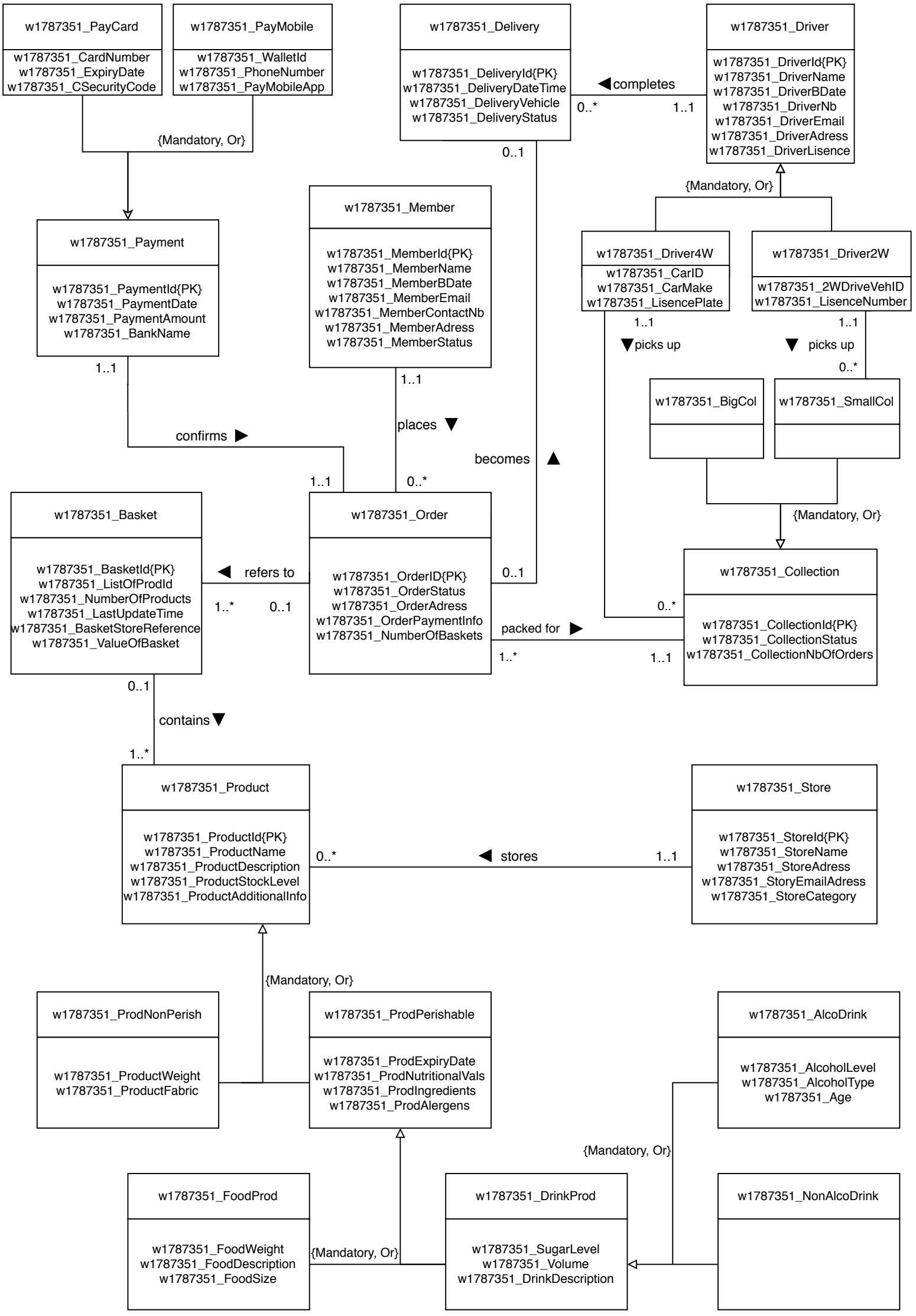
Name : Franciszek Ruszkowski
Student number : w1787351
Course : Data Science And Analytics
Coursework Part A + B
22.11.2020
Tutorial Group : Reni Saumya, Chountas Panagiotis Tuesday


w1787351_PayCard
w1787351_CardNumber
w1787351_ExpiryDate
w1787351_CSecurityCode
w1787351_PayMobile
w1787351_WalletId
w1787351_PhoneNumber
w1787351_PayMobileApp
{Mandatory, Or}
w1787351_Payment
w1787351_PaymentId{PK}
w1787351_PaymentDate
w1787351_PaymentAmount
w1787351_BankName
1..1
confirms
w1787351_Delivery
w1787351_DeliveryId{PK}
w1787351_DeliveryDateTime
w1787351_DeliveryVehicle
w1787351_DeliveryStatus
completes
0..*
1..1
0..1
w1787351_Driver
w1787351_DriverId{PK}
w1787351_DriverName
w1787351_DriverBDate
w1787351_DriverNb
w1787351_DriverEmail
w1787351_DriverAdress
w1787351_DriverLisence
{Mandatory, Or}
w1787351_Member
w1787351_MemberId{PK}
w1787351_MemberName
w1787351_MemberBDate
w1787351_MemberEmail
w1787351_MemberContactNb
w1787351_MemberAdress
w1787351_MemberStatus
1..1
places
0..*
w1787351_Driver4W
w1787351_CarID
w1787351_CarMake
w1787351_LisencePlate
w1787351_Driver2W
w1787351_2WDriveVehID
w1787351_LisenceNumber
1..1
picks up
1..1
picks up
0..*
w1787351_BigCol
w1787351_SmallCol
becomes
{Mandatory, Or}
w1787351_Basket
w1787351_BasketId{PK}
w1787351_ListOfProdId
w1787351_NumberOfProducts
w1787351_LastUpdateTime
w1787351_BasketStoreReference
w1787351_ValueOfBasket
refers to
1..*
0..1
w1787351_Order
w1787351_OrderID{PK}
w1787351_OrderStatus
w1787351_OrderAdress
w1787351_OrderPaymentInfo
w1787351_NumberOfBaskets
0..1
w1787351_Collection
w1787351_CollectionId{PK}
w1787351_CollectionStatus
w1787351_CollectionNbOfOrders
0..*
packed for
1..*
1..1
0..1
contains
1..*
w1787351_Product
w1787351_ProductId{PK}
w1787351_ProductName
w1787351_ProductDescription
w1787351_ProductStockLevel
w1787351_ProductAdditionalInfo
0..*
stores
1..1
w1787351_Store
w1787351_StoreId{PK}
w1787351_StoreName
w1787351_StoreAdress
w1787351_StoryEmailAdress
w1787351_StoreCategory
{Mandatory, Or}
w1787351_ProdNonPerish
w1787351_ProductWeight
w1787351_ProductFabric
w1787351_ProdPerishable
w1787351_ProdExpiryDate
w1787351_ProdNutritionalVals
w1787351_ProdIngredients
w1787351_ProdAlergens
w1787351_AlcoDrink
w1787351_AlcoholLevel
w1787351_AlcoholType
w1787351_Age
{Mandatory, Or}
w1787351_FoodProd
w1787351_FoodWeight
w1787351_FoodDescription
w1787351_FoodSize
{Mandatory, Or}
w1787351_DrinkProd
w1787351_SugarLevel
w1787351_Volume
w1787351_DrinkDescription
w1787351_NonAlcoDrink

Data Dictionary on Entities

Entity Name	Description
w1787351_Member	FTY Member Details
w1787351_Delivery	Delivery details
w1787351_Order	The order details confirmed by member
w1787351_Store	Details on data regarding the Stores register by FTY

w1787351_Payment	w1787351_PayCard	A payment has to be made either
	w1787351_PayMobile	by card of by phone
w1787351_Driver	w1787351_Driver4W	A driver has to drive either a 4 wheel
	w1787351_Driver2W	vehichle or 2 wheel vehichle
w1787351_Collection	w1787351_BigCol	A collection has to be either big
	w1787351_SmallCol	or has to be small
w1787351_Product	w1787351_ProdPerishable	A product has to be either Perishabl
	w1787351_ProdNonPerish	or non perishable
w1787351_ProdPerishable	w1787351_DrinkProd	Perishable products have to drinks
	w1787351_FoodProd	or food
w1787351_DrinkProd	w1787351_AlcoDrink	Drinks have to be either alcoholic
	w1787351_NonAlcoDrink	or non alcoholic

Data Dictionarh on relationships

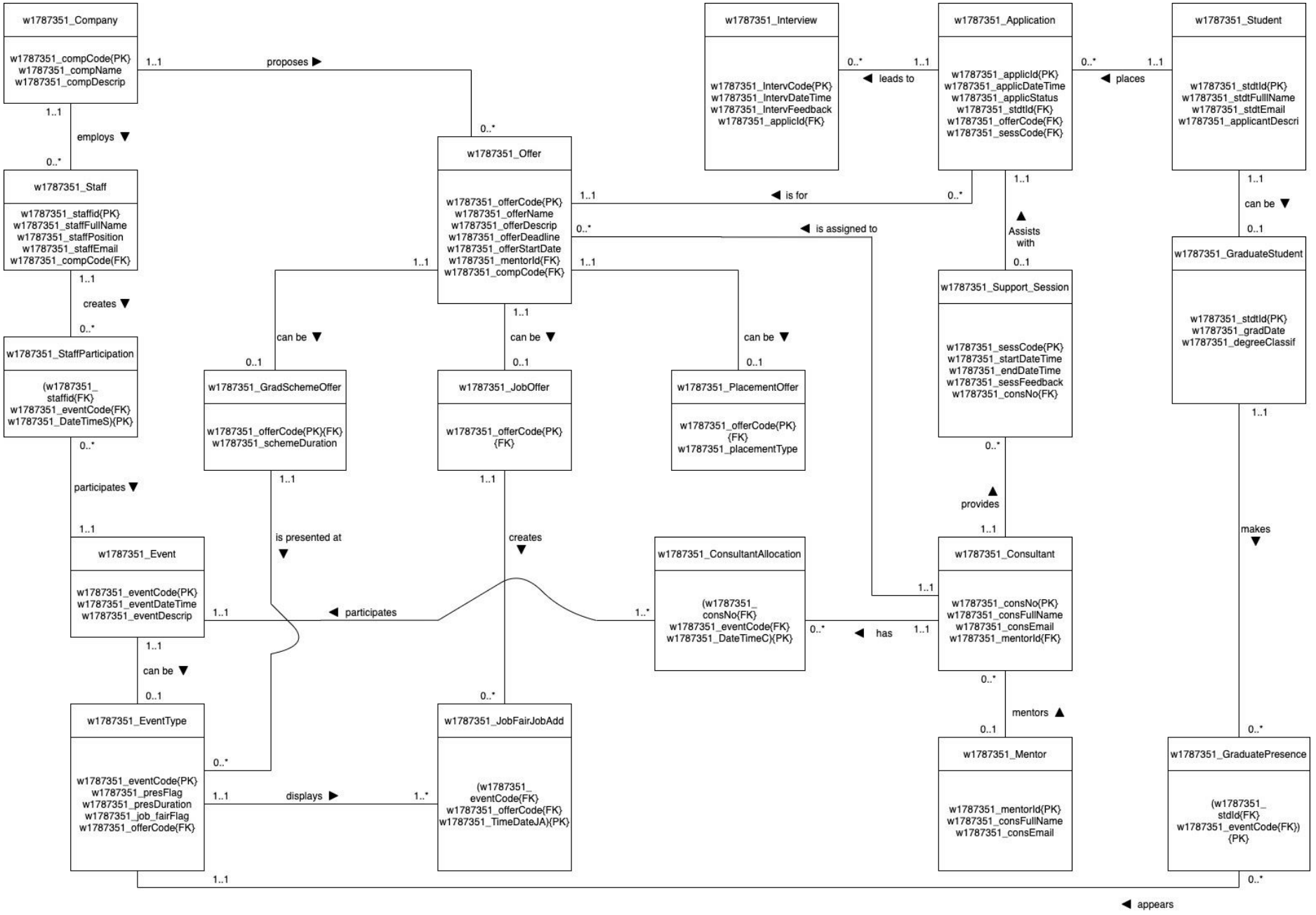
w1787351_Member	1..1	Places	0..*	w1787351_Order	Member may not have confirmed an order
					Many order can be made
					There is no order without member creating it
					An order has maximum one member
w1787351_Payment	1..1	Confirms	1..1	w1787351_Order	An order doesnt exist yet without a payment
					Order refers to max one payment
					Payment confirms an order
					A payment cannot confirm more than 1 order
w1787351_Driver	1..1	Completes	0..*	w1787351_Delivery	Delivery does not exist without a driver
					One delivery has max 1 driver
					New member - no drivers delivered anything
					Many different drivers can deliver
w1787351_Order	1..*	Packed for	1..1	w1787351_Collection	An order has to be packed for collection
					An order packed for max 1 collection
					A collection doesn't exist without at least 1 order
					A collection can have many orders inside
w1787351_Order	0..1	Contains	1..*	w1787351_Product	An order cannot be empty
					An order can contain many products
					A product may not be assigned to order
					A product can be in maximum one order
w1787351_Basket	1..1	Refers to	0..*	w1787351_Store	A basket refers to one store
					A basket cannot refer to more than one stores
					A store can not have any baskets yet
					A store can have multiple baskets refering to it
w1787351_Store	1..1	Stores	0..*	w1787351_Product	A new store, 0 products so far
					A store can store many products
					A product cannot be without a store
					A product can be in maximum 1 store

w1787351_Driver4W	1..1	Picks up	0..*	w1787351_Collection	New Driver, may have 0 collections picked up
					A driver can have many collections
					A collection has to be assigned to a driver
					A collection cannot be assigned to >1 driver
w1787351_Driver2W	1..1	Picks up	0..*	w1787351_SmallCol	New Driver, may have 0 collections picked up
					A driver can have many collections
					A collection has to be assigned to a driver
					A collection cannot be assigned to >1 driver

Data Dictionary on Atrributes

w1787351_Member	
w1787351_MemberId{PK} w1787351_MemberName w1787351_MemberBDate w1787351_MemberEmail w1787351_MemberContactNb w1787351_MemberAdress w1787351_MemberStatus	Unique Id number of member - therefore PK Member's full name Member's birth date Member's Email address Member's telephone contact phone number Member's residential address (for contact) Member's verification status (new,pending)
w1787351_Order	
w1787351_OrderID{PK} w1787351_OrderStatus w1787351_OrderAdress w1787351_OrderPaymentInfo w1787351_NumberOfBaskets	Unique Order Id number - therefore PK Order status (payed for, completed) Order address - delivery address Order payment info - method of payment etc Number of baskets
w1787351_Basket	
w1787351_BasketId{PK} w1787351_ListOfProdId w1787351_NumberOfProducts w1787351_LastUpdateTime w1787351_BasketStoreReference w1787351_ValueOfBasket	Unique basket Id number - hence PK Listed products in a basket Number of Products Last Update of basket date/time Reference number to store Monetary value of basket
w1787351_Payment	
w1787351_PaymentId{PK} w1787351_PaymentDate w1787351_PaymentAmount w1787351_BankName	Unique Payment Id number - hence PK Payment date and time Payment monetary amount Bank's name from where paid
w1787351_Delivery	
w1787351_DeliveryId{PK} w1787351_DeliveryDateTime w1787351_DeliveryVehicle w1787351_DeliveryStatus	Delivery Id number unique - hence PK Delivery Date and time Type of vehicle that delivered - 4w or 2w Delivery status - pending, completed etc
w1787351_Store	
w1787351_StoreId{PK} w1787351_StoreName w1787351_StoreAdress w1787351_StoreEmailAdress w1787351_StoreCategory	Store's unique Id number - hence PK Store's official name Store's location address Store's email address Store's category of products they sell
w1787351_ProdNonPerish	
w1787351_ProductWeight w1787351_ProductFabric	Non perishable product's weight Non perishable product's fabrtic

w1787351_PayCard	
w1787351_CardNumber w1787351_ExpiryDate w1787351_CSecurityCode	Credit/Debit card number Credit/Debit card expiry date CSV card security number
w1787351_PayMobile	
w1787351_WalletId w1787351_PhoneNumber w1787351_PayMobileApp	MobileWallet Identification number Telephone number conected to wallet Apps name for phone payment method
w1787351_Driver	
w1787351_DriverId{PK} w1787351_DriverName w1787351_DriverBDate w1787351_DriverNb w1787351_DriverEmail w1787351_DriverAdress w1787351_DriverLisence	Driver's unique Id number- hence PK Driver's full name Driver's date of birth Driver's telephone contact number Driver's email address Driver's home address Driver's driver lisence serial number
w1787351_Driver4W	
w1787351_CarID w1787351_CarMake w1787351_LisencePlate	Car id number when registered for foodtooyou Car's make Car's Lisence Plate
w1787351_Driver2W	
w1787351_2WDriveVehID w1787351_LisenceNumber	2 wheel vehicle id number when registered 2 wheel vehicle lisence number
w1787351_Collection	
w1787351_CollectionId{PK} w1787351_CollectionStatus w1787351_CollectionNbOfOrders	Collection's Id unique number - PK Collection status - pending, done Number of orders in a collection
w1787351_Product	
w1787351_ProductId{PK} w1787351_ProductName w1787351_ProductDescription w1787351_ProductStockLevel w1787351_ProductAdditionalInfo	Product's unique id number - PK Product's original name Product's description Product's stock level Product's additional information
w1787351_ProdPerishable	
w1787351_ProdExpiryDate w1787351_ProdNutritionalVals w1787351_ProdIngredients w1787351_ProdAlergens	Product's expiry date Product's nutritional values Product's ingredienrts Product's alergens
w1787351_AlcoDrink	
w1787351_AlcoholLevel w1787351_AlcoholType w1787351_Age	Alcoholic drink's alcohol level Alcoholic drink's alcohol type Alcoholic drink's age


w1787351_Company
w1787351_compCode{PK}
w1787351_compName
w1787351_compDescrip
proposes
employs
w1787351_Staff
w1787351_staffid{PK}
w1787351_staffFullName
w1787351_staffPosition
w1787351_staffEmail
w1787351_compCode{FK}
creates
w1787351_StaffParticipation
(w1787351_
staffid{FK}
w1787351_eventCode{FK}
w1787351_DateTimeS){PK}
participates
w1787351_Event
w1787351_eventCode{PK}
w1787351_eventDateTime
w1787351_eventDescrip
can be
w1787351_EventType
w1787351_eventCode{PK}
w1787351_presFlag
w1787351_presDuration
w1787351_job_fairFlag
w1787351_offerCode{FK}
displays
w1787351_Offer
w1787351_offerCode{PK}
w1787351_offerName
w1787351_offerDescrip
w1787351_offerDeadline
w1787351_offerStartDate
w1787351_mentorId{FK}
w1787351_compCode{FK}
is for
is assigned to
can be
w1787351_GradSchemeOffer
w1787351_offerCode{PK}{FK}
w1787351_schemeDuration
can be
w1787351_JobOffer
w1787351_offerCode{PK}
{FK}
can be
w1787351_PlacementOffer
w1787351_offerCode{PK}
{FK}
w1787351_placementType
is presented at
creates
w1787351_JobFairJobAdd
(w1787351_
eventCode{FK}
w1787351_offerCode{FK}
w1787351_TimeDateJA){PK}
participates
w1787351_ConsultantAllocation
(w1787351_
consNo{FK}
w1787351_eventCode{FK}
w1787351_DateTimeC){PK}
has
w1787351_Interview
w1787351_IntervCode{PK}
w1787351_IntervDateTime
w1787351_IntervFeedback
w1787351_applicId{FK}
leads to
w1787351_Application
w1787351_applicId{PK}
w1787351_applicDateTime
w1787351_applicStatus
w1787351_stdtId{FK}
w1787351_offerCode{FK}
w1787351_sessCode{FK}
places
Assists
with
w1787351_Support_Session
w1787351_sessCode{PK}
w1787351_startDateTime
w1787351_endDateTime
w1787351_sessFeedback
w1787351_consNo{FK}
provides
w1787351_Consultant
w1787351_consNo{PK}
w1787351_consFullName
w1787351_consEmail
w1787351_mentorId{FK}
mentors
w1787351_Mentor
w1787351_mentorId{PK}
w1787351_consFullName
w1787351_consEmail
w1787351_Student
w1787351_stdtId{PK}
w1787351_stdtFullName
w1787351_stdtEmail
w1787351_applicantDescri
can be
w1787351_GraduateStudent
w1787351_stdtId{PK}
w1787351_gradDate
w1787351_degreeClassif
makes
w1787351_GraduatePresence
(w1787351_
stdId{FK}
w1787351_eventCode{FK})
{PK}
appears

- We first consider the specialisations and 1:1 mandatory on both sides.
- There are no 1:1 mandatory on both sides so we just map the specialisations

1. Student entity with {optional} specialisation to Graduate. We apply rule 10, so that it becomes 2 tables : w1787351_Student and w1787351_GraduateStudent, multiplicities 1..1 and 0..1 respectively, and the relationship "Can Be". Having 1 to 1 optional, we copy the primary key from the w1787351_Student (Parent) table and put it as the foreign key in the w1787351_GraduateStudent (Child) table, along with all the attributes that were in the Graduate specialisation before.
2. Event entity with an {optional - and} specialisation. We apply rule 8, so that it becomes 2 tables : w1787351_Event and w1787351_EventType, multiplicities 1...1 and 0..1 respectively, and the relationship "Can Be". Having 1 to 1 optional, we copy the primary key from the w1787351_Event (Parent) table and put it as the foreign key in the w1787351_EventType (Child) table, and further following rule 8 - we add flag attributes, along with all the attributes that were in the specialisations before.
3. Offer entity with an {optional - or} specialisation. We apply rule 10, so that it becomes 4 tables : w1787351_Offer, w1787351_GradSchemeOffer, w1787351_JobOffer and w1787351_PlacementOffer, multiplicities 1...1 at Offer and 0..1 on all new tables, and all the relationships become "Can Be". Having 1 to 1 optional, we copy the primary key from the w1787351_Offer (Parent) table and put it as the foreign key in all new (Child) tables, along with all the attributes that were in the Offer specialisations before.

- We now consider many to many relationships, and any ternary relationships
- There are no ternary relationships so we just map the many to many relationships

4. Staff and Event entities have a many to many relationship. We use rule 5 and create 3 tables. 2 Parent tables (w1787351_Staff and w1787351_Event) linked with a new child table - w1787351_StaffParticipation. Both parent tables have a 1:M relationship with the newly created child table. The Primary key in the Child table is a combination of 2 Primary Keys of the parent tables, which are now foreign keys in the Child table. The Primary Key in the child table is composite, because a Staff can attend several events, therefore we add the w1787351_DateTimeS. The new relationships are the following: w1787351_Staff creates w1787351_StaffParticipation, and w1787351_StaffParticipation participates in w1787351_Event.
5. Consultant and Event entities have a many to many relationship. We use rule 5 and create 3 tables. 2 Parent tables (w1787351_Consultant and w1787351_Event) linked with a new child table - w1787351_ConsultantAllocation. Both parent tables have a 1:M relationship with the newly created child table. The Primary key in the Child table is a combination of 2 Primary Keys of the parent tables, which are now foreign keys in the Child table. The Primary Key in the child table is composite, because a Consultant can be allocated to several events, therefore we add the w1787351_DateTimeC. The new relationships are the following: w1787351_Consultant has w1787351_ConsultantAllocation, and w1787351_ConsultantAllocation participates in w1787351_Event.
6. JobOffer and EventType tables have a many to many relationship. We use rule 5 and create 3 tables. 2 Parent tables (w1787351_JobOffer and w1787351_EventType) linked with a new

child table - w1787351_JobFairJobAdd. Both parent tables have a 1:M relationship with the newly created child table. The Primary key in the Child table is a combination of 2 Primary Keys of the parent tables, which are now foreign keys in the Child table. The Primary Key in the child table is composite, because a JobFairJobAdd the same Add can be displayed at several events, therefore we add the w1787351_TimeDateJA. The new relationships are the following: w1787351_JobOffer creates w1787351_JobFairJobAdd and w1787351_EventType displays w1787351_JobOfferJobAdd

7. GraduateStudent and EventType tables have a many to many relationship. We use rule 5 and create 3 tables. 2 Parent tables (w1787351_GraduateStudent and w1787351_EventType) linked with a new child table - w1787351_GraduatePresence. Both parent tables have a 1:M relationship with the newly created child table. The Primary key in the Child table is a combination of 2 Primary Keys of the parent tables, which are now foreign keys in the Child table. The Primary Key in the child table is compound, because a GraduateStudent cannot be at two events at the same time. The new relationships are the following: w1787351_GraduateStudent creates w1787351_GraduatePresence and w1787351_GraduatePresence appears at w1787351_EventType.

- We now map the 1.1 relationships optional on one side/ optional on both sides - and the 1 to many

8. Student and Application entities have a 1 to many relationship. To map them to a Logical ERD we have to use rule number 1. The entity with 0..1 multiplicity becomes the Parent Table and entity with 0..* multiplicity becomes the Child table. The primary key from the Parent Table is copied as a foreign key to the Child Table, apart from that all the attributes remain the same in the new tables, The relationship stays the same - w1787351_Student places w1787351_Application.
9. Interview and Application entities have a 1 to many relationship. To map them to a Logical ERD we have to use rule number 1. The entity with 0..1 multiplicity becomes the Parent Table and entity with 0..* multiplicity becomes the Child table. The primary key from the Parent Table is copied as a foreign key to the Child Table, apart from that all the attributes remain the same in the new tables, The relationship stays the same - w1787351_Application leads to an w1787351_Interview.
10. Offer and Application entities have a 1 to many relationship. To map them to a Logical ERD we have to use rule number 1. The entity with 0..1 multiplicity becomes the Parent Table and entity with 0..* multiplicity becomes the Child table. The primary key from the Parent Table is copied as a foreign key to the Child Table, apart from that all the attributes remain the same in the new tables, The relationship stays the same - w1787351_Application is for an w1787351_Offer
11. Consultant is an entity that has a relationship with itself - “Mentors”. However, we still use rule number 1, as there is a 1 to many relationship. Therefore we get a table w1787351_Consultant we introduce a FK in the same table with a different name (w1787351_mentorId) to reference the PK of the same table.
12. Consultant and Support_Session entities have a 1 to many relationship. To map them to a Logical ERD we have to use rule number 1. The entity with 0..1 multiplicity becomes the Parent Table and entity with 0..* multiplicity becomes the Child table. The primary key from

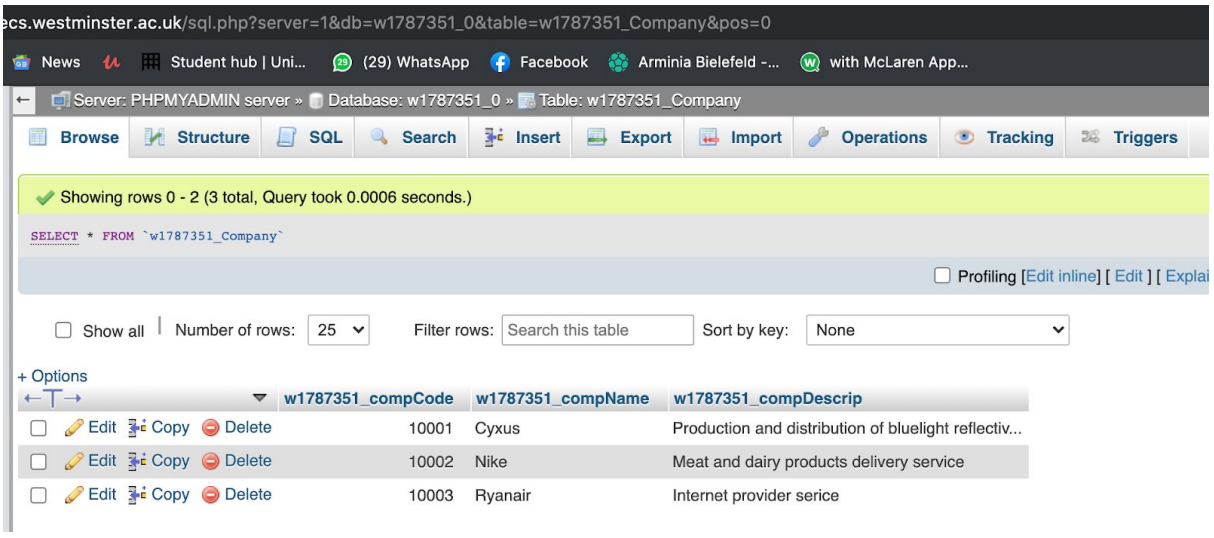
the Parent Table is copied as a foreign key to the Child Table, apart from that all the attributes remain the same in the new tables, The relationship stays the same - w1787351_Consultant provides w1787351_Support_Session

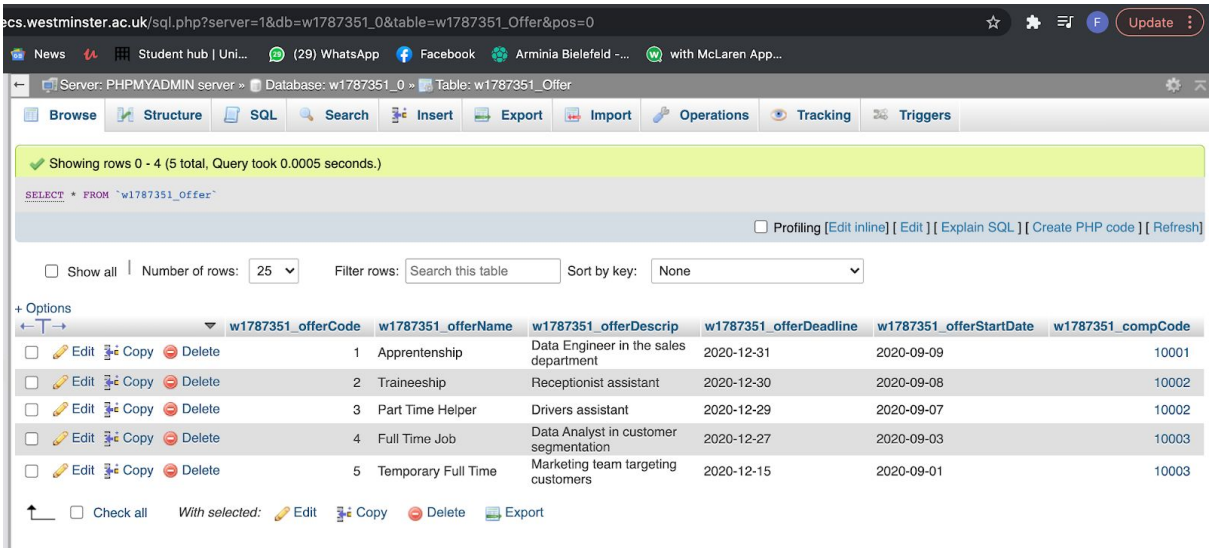
13. Support_Session and Application entities have a 1 to 1 optional on one side relationship. To map them, we have to use rule number 3, which tells us to create 2 tables with the same, and decide which one is “more mandatory” and could be the Parent Table. In this case w1787351_Application will become the Parent Table, w1787351_Support_Session will become the Child table, the multiplicities remain the same and the relationship remains “assists with”. The only addition is now the Child table has the Primary key of the parent table as the foreign key
14. Offer and Consultant entities have a 1 to many relationship. To map them to a Logical ERD we have to use rule number 1. The entity with 0..1 multiplicity becomes the Parent Table and entity with 0..* multiplicity becomes the Child table. The primary key from the Parent Table is copied as a foreign key to the Child Table, apart from that all the attributes remain the same in the new tables, The relationship stays the same - w1787351_Consultant is assigned to an w1787351_Offer.
15. Offer and Company entities have a 1 to many relationship. To map them to a Logical ERD we have to use rule number 1. The entity with 0..1 multiplicity becomes the Parent Table and entity with 0..* multiplicity becomes the Child table. The primary key from the Parent Table is copied as a foreign key to the Child Table, apart from that all the attributes remain the same in the new tables, The relationship stays the same - w1787351_Company proposes an w1787351_Offer.
16. Staff and Company entities have a 1 to many relationship. To map them to a Logical ERD we have to use rule number 1. The entity with 0..1 multiplicity becomes the Parent Table and entity with 0..* multiplicity becomes the Child table. The primary key from the Parent Table is copied as a foreign key to the Child Table, apart from that all the attributes remain the same in the new tables, The relationship stays the same - w1787351_Company employs w1787351_Staff.
17. Staff and Company entities have a 1 to many relationship. To map them to a Logical ERD we have to use rule number 1. The entity with 0..1 multiplicity becomes the Parent Table and entity with 0..* multiplicity becomes the Child table. The primary key from the Parent Table is copied as a foreign key to the Child Table, apart from that all the attributes remain the same in the new tables, The relationship stays the same - w1787351_Company employs w1787351_Staff.
18. GradScheme and EventType have a 1 to many relationship. To map them to a Logical ERD we have to use rule number 1. The entity with 0..1 multiplicity becomes the Parent Table and entity with 0..* multiplicity becomes the Child table. The primary key from the Parent Table is copied as a foreign key to the Child Table, apart from that all the attributes remain the same in the new tables, The relationship stays the same - w1787351_GradScheme is presented at w1787351_EventType.

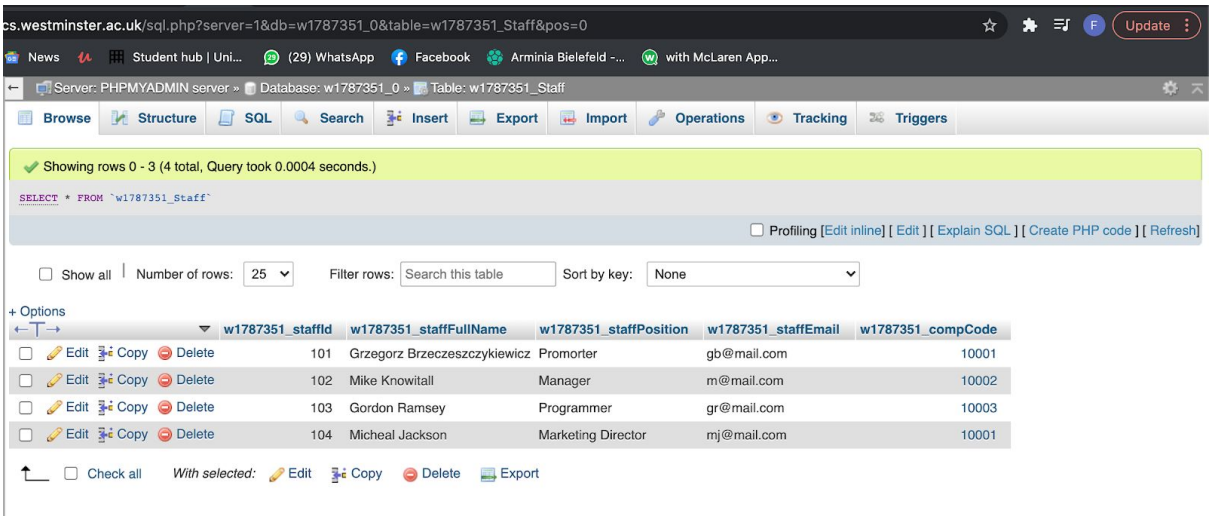
Question 7

```sql
DROP TABLE IF EXISTS Offer;
DROP TABLE IF EXISTS Staff;
DROP TABLE IF EXISTS Company;
DROP TABLE IF EXISTS w1787351_Offer;
DROP TABLE IF EXISTS w1787351_Staff;
DROP TABLE IF EXISTS w1787351_Company;

-- Question 7

-- Create the w1787351_Company table

CREATE TABLE w1787351_Company
(
    w1787351_compCode       INT(5),
    w1787351_compName       VARCHAR(50) UNIQUE NOT NULL,
    w1787351_compDescrip     VARCHAR(200) NOT NULL,
    constraint      w1787351_c_ccode_pk PRIMARY KEY (w1787351_compCode)
);

-- Create the w1787351_Staff Table

CREATE TABLE w1787351_Staff
(
    w1787351_staffId         INT(5),
    w1787351_staffFullName   VARCHAR(50) NOT NULL,
    w1787351_staffPosition   VARCHAR(50),
    w1787351_staffEmail      VARCHAR(50) NOT NULL unique,
    w1787351_compCode        INT(5) NOT NULL,
    constraint      w1787351_s_sid_pk PRIMARY KEY (w1787351_staffId),
    constraint      w1787351_s_ccode_fk  FOREIGN KEY (w1787351_compCode)
    references      w1787351_Company(w1787351_compCode)
);

-- Create the w1787351_Offer Table

CREATE TABLE w1787351_Offer
(
    w1787351_offerCode      INT(5),
    w1787351_offerName      VARCHAR(50) NOT NULL ,
    w1787351_offerDescrip   VARCHAR(200) NOT NULL,
    w1787351_offerDeadline  DATE,
    w1787351_offerStartDate DATE,
    w1787351_compCode       INT(5) NOT NULL,
    constraint     w1787351_o_ocode_pk PRIMARY KEY (w1787351_offerCode),
    constraint     w1787351_o_ccode_fk  FOREIGN KEY (w1787351_compCode)
    references     w1787351_Company(w1787351_compCode)
);
```

```sql
);

-- Create the w1787351_Offer Table

CREATE TABLE w1787351_Offer
(
    w1787351_offerCode      INT(5),
    w1787351_offerName      VARCHAR(50) NOT NULL ,
    w1787351_offerDescrip   VARCHAR(200) NOT NULL,
    w1787351_offerDeadline  DATE,
    w1787351_offerStartDate DATE,
    w1787351_compCode       INT(5) NOT NULL,
    constraint     w1787351_o_ocode_pk PRIMARY KEY (w1787351_offerCode),
    constraint     w1787351_o_ccode_fk  FOREIGN KEY (w1787351_compCode)
    references     w1787351_Company(w1787351_compCode)
);

-- Input the data into Company

INSERT INTO
w1787351_Company (w1787351_compCode, w1787351_compName, w1787351_compDescrip)
VALUES
(10001,'Cyxus','Production and distribution of bluelight reflective glasses'),
(10002,'Nike','Meat and dairy products delivery service'),
(10003,'Ryanair','Internet provider serice');

-- Input the data into Staff

INSERT INTO
w1787351_Staff (w1787351_staffId, w1787351_staffFullName,
 w1787351_staffPosition,w1787351_staffEmail,w1787351_compCode)
VALUES
(101, 'Grzegorz Brzeczeszczykiewicz', 'Promorter','gb@mail.com', 10001),
(102, 'Mike Knowitall', 'Manager','m@mail.com', 10002),
(103, 'Gordon Ramsey', 'Programmer','gr@mail.com', 10003),
(104, 'Micheal Jackson', 'Marketing Director ','mj@mail.com', 10001);

-- Input the data into Offer

INSERT INTO
w1787351_Offer (w1787351_offerCode, w1787351_offerName, w1787351_offerDescrip,
 w1787351_offerDeadline, w1787351_offerStartDate, w1787351_compCode)
VALUES
(1,'Apprentenship','Data Engineer in the sales department','2020-12-31','2020-09-09',10001),
(2,'Traineeship','Receptionist assistant','2020-12-30','2020-09-08',10002),
(3,'Part Time Helper','Drivers assistant' ,'2020-12-29','2020-09-07',10002),
(4,'Full Time Job','Data Analyst in customer segmentation','2020-12-27','2020-09-03',10003),
(5,'Temporary Full Time','Marketing team targeting customers','2020-12-15','2020-09-01',10003);
```

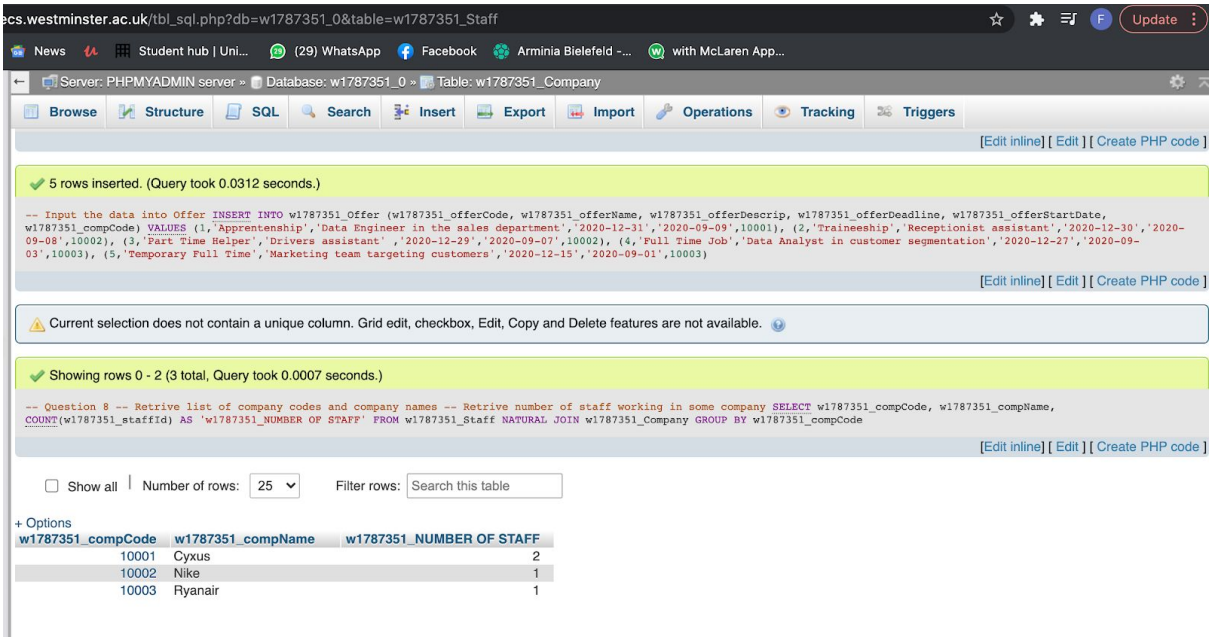

ecs.westminster.ac.uk/sql.php?server=1&db=w1787351_0&table=w1787351_Company&pos=0
News
Student hub | Uni...
(29) WhatsApp
Facebook
Arminia Bielefeld -...
with McLaren App...
Server: PHPMYADMIN server » Database: w1787351_0 » Table: w1787351_Company
Browse
Structure
SQL
Search
Insert
Export
Import
Operations
Tracking
Triggers
Showing rows 0 - 2 (3 total, Query took 0.0006 seconds.)
SELECT * FROM `w1787351_Company`
Profiling [Edit inline] [Edit] [Explai
Show all
Number of rows: 25
Filter rows: Search this table
Sort by key: None
+ Options
w1787351_compCode
w1787351_compName
w1787351_compDescrip
Edit Copy Delete 10001 Cyxus Production and distribution of bluelight reflectiv...
Edit Copy Delete 10002 Nike Meat and dairy products delivery service
Edit Copy Delete 10003 Ryanair Internet provider serice


ecs.westminster.ac.uk/sql.php?server=1&db=w1787351_0&table=w1787351_Offer&pos=0
Update
Server: PHPMYADMIN server » Database: w1787351_0 » Table: w1787351_Offer
Browse
Structure
SQL
Search
Insert
Export
Import
Operations
Tracking
Triggers
Showing rows 0 - 4 (5 total, Query took 0.0005 seconds.)
SELECT * FROM `w1787351_Offer`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all
Number of rows: 25
Filter rows: Search this table
Sort by key: None
+ Options
w1787351_offerCode
w1787351_offerName
w1787351_offerDescrip
w1787351_offerDeadline
w1787351_offerStartDate
w1787351_compCode
Edit Copy Delete 1 Apprentenship Data Engineer in the sales department 2020-12-31 2020-09-09 10001
Edit Copy Delete 2 Traineeship Receptionist assistant 2020-12-30 2020-09-08 10002
Edit Copy Delete 3 Part Time Helper Drivers assistant 2020-12-29 2020-09-07 10002
Edit Copy Delete 4 Full Time Job Data Analyst in customer segmentation 2020-12-27 2020-09-03 10003
Edit Copy Delete 5 Temporary Full Time Marketing team targeting customers 2020-12-15 2020-09-01 10003
Check all
With selected:
Edit
Copy
Delete
Export


cs.westminster.ac.uk/sql.php?server=1&db=w1787351_0&table=w1787351_Staff&pos=0
Update
Server: PHPMYADMIN server » Database: w1787351_0 » Table: w1787351_Staff
Browse
Structure
SQL
Search
Insert
Export
Import
Operations
Tracking
Triggers
Showing rows 0 - 3 (4 total, Query took 0.0004 seconds.)
SELECT * FROM `w1787351_Staff`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all
Number of rows: 25
Filter rows: Search this table
Sort by key: None
+ Options
w1787351_staffId
w1787351_staffFullName
w1787351_staffPosition
w1787351_staffEmail
w1787351_compCode
Edit Copy Delete 101 Grzegorz Brzeczeszczykiewicz Promorter gb@mail.com 10001
Edit Copy Delete 102 Mike Knowitall Manager m@mail.com 10002
Edit Copy Delete 103 Gordon Ramsey Programmer gr@mail.com 10003
Edit Copy Delete 104 Micheal Jackson Marketing Director mj@mail.com 10001
Check all
With selected:
Edit
Copy
Delete
Export

Question 8

```sql
-- Question 8

-- Retrive list of company codes and company names
-- Retrive number of staff working in some company

SELECT w1787351_compCode, w1787351_compName, COUNT(w1787351_staffId) AS 'w1787351_NUMBER OF STAFF'
FROM w1787351_Staff NATURAL JOIN w1787351_Company
GROUP BY w1787351_compCode;
```



Question 9

```sql

-- Question 9

-- Write a SQL query that displays a list of company names along
-- the names and positions of staff they employ
-- and the names and descriptions of the offers they propose.

SELECT w1787351_compName, w1787351_staffFullName,
w1787351_staffPosition, w1787351_offerName, w1787351_offerDescrip
FROM w1787351_Company c
JOIN w1787351_Staff s
ON s.w1787351_compCode = c.w1787351_compCode
JOIN w1787351_Offer o
ON o.w1787351_compCode = c.w1787351_compCode
ORDER BY w1787351_compName;

```