ECU178 Computer Science:
220CT Data and Information Retrieval
Coursework

Due on March 13th 2015

Robert Rigler : 4939377

Contents

Task 1 Database Design

This task in Database Design consists of four activities. The first three involve normalising the given data to third normal form, and the fourth is to produce and Entity Relationship diagram of the normalised relations. For each activity I will give a detailed step by step explanation of how I completed each activity.

Activity 1: First Normal Form

To put this data into First Normal Form (1NF), I need to:

1. Identify any repeating redundant data and remove it from the current Entity

2. Place the data into a new Entity

3. Create a relationship with a primary key from one Entity as a foreign key in the other.

Step 1: Identify Redundancy

On inspecting the data, I can see that there are multiple instances of repeating data.
Orders' ID: *CON-2237, CON-2356* and *CON-1234* all have repeating data entries for fields: *Equipment, Qty,* and *Unit Price* .

Step 2: Create New Entity

Removing the *Equipment, Qty,* and *Unit Price* fields and placing them in a new entity, leaves me with two entities as shown below.

Step 3: Relationships and Keys

To complete the First Normal Form, a relationship needs to be created between the entities.
I created the relationship by including the *Order ID* attribute as a foreign key in the *ItemOrder* entity.
Order ID is used as a Primary Key for the *Order* entity. In the *ItemOrder* entity, no one attribute can be used to uniquely identify a single record. For this reason, I have created a concatenated key using the attributes *Equipment* and *Order ID*. The concatenated key can now be used to uniquely identify each record.

1NF : Diagram

Order	(Order ID		ItemOrder	(*Order ID
	Supplier ID			Equipment
	Client Name			Qty)
	Client Address			Unit Price)
	Date			
	Total Price)			

1NF : Data

Order entity

Order ID	Supplier ID	Client Name	Client Address	Date	Total Price
CON-2237	168	Coventry Building Services Ltd	Units 2-4, Binley Industrial Estate, CV3 2WL	14-Dec-14	£99.00
CON-3664	527	Allied Construction Ltd	34,Lythalls La Industrial Estate, CV6 6RG	16-Jan-15	£36.00
CON-2356	169	Rioh Builds Ltd	Unit 12,Stoneleigh Park, CV8 2UV	12-Feb-15	£280.00
CON-1234	032	Grand Design Ltd	32-34, Bilton Industrial Estate, CV3 5YB	16-Apr-15	£23.00

ItemOrder entity

Order ID	Equipment	Qty	Unit Price
CON-2237	Butterfly Valve	2	£5.00
CON-2237	3/4" Locknut	6	£1.50
CON-2237	Sch 40 Blk Pipe	4	£20.00
CON-3664	Thin Stranded Copper Wire	6	£6.00
CON-2356	Sch 40 Blk Pipe	3	£20.00
CON-2356	4x8x3/4 Cos Plywood	2	£10.00
CON-2356	3/4" EMT	2	£50.00
CON-2356	Duplex Ivy Rec	1	£100.00
CON-1234	Sch 40 Blk Pipe	1	£20.00
CON-1234	3/4" Locknut	2	£1.50

Activity 2: Second Normal Form

To put this data into the Second Normal Form(2NF) I need to ensure that the attributes are completely dependant on the primary key, i.e., that no attribute is only dependant on one part of the primary key. This can be done in two steps:

1. Test each attribute for complete dependency on the primary key.

2. Remove any partially dependent attributes to a new entity and assign a primary key.

For this particular set of data, the *Order* entity does not have a concatenated key and therefore is already in the second normal form.

Step 1: Testing each attribute.

Primary Key	Attribute	Functionally Dependant?
Order ID, Equipment	Qty	Yes, dependant on both
Order ID, Equipment	Unit Price	No, dependant on Equipment only

Step 2: Entity, Relationship and Key

From my testing, I found that the attribute *Unit Price* is not functionally dependant as it is only dependent on *Equipment*, but not *Order ID*.
I moved *Equipment* and *Unit Price* into a new entity called *Item* and made *Equipment* the primary key.
I created a relationship between the *Item* and *ItemOrder* entities by repeating the *Equipment* attribute in *ItemOrder* as a foreign key.

Entities in Second Normal Form: Diagram

Order	(Order ID	**Item**	(Equipment	**ItemOrder**	(*Order ID
	Suppler ID		Unit Price)		*Equipment
	Client Name				Qty)
	Client Address				
	Date				
	Total Price)				

Entities in Second Normal Form: With data

Order entity

Order ID	Supplier ID	Client Name	Client Address	Date	Total Price
CON-2237	168	Coventry Building Services Ltd	Units 2-4, Binley Industrial Estate, CV3 2WL	14-Dec-14	£99.00
CON-3664	527	Allied Construction Ltd	34,Lythalls La Industrial Estate, CV6 6RG	16-Jan-15	£36.00
CON-2356	169	Rioh Builds Ltd	Unit 12,Stoneleigh Park, CV8 2UV	12-Feb-15	£280.00
CON-1234	032	Grand Design Ltd	32-34, Bilton Industrial Estate, CV3 5YB	16-Apr-15	£23.00

Item entity

Equipment	Unit Price
3/4" EMT	50.00
3/4" Locknut	1.50
4x8x3/4 Cos Plywood	10.00
Butterfly Valve	5.00
Duplex Ivy Rec	100.00
Sch 40 Blk Pipe	20.00
Thin Stranded Copper Wire	6.00

ItemOrder entity

Order ID	Equipment	Qty	Unit Price
CON-2237	Butterfly Valve	2	£5.00
CON-2237	3/4" Locknut	6	£1.50
CON-2237	Sch 40 Blk Pipe	4	£20.00
CON-3664	Thin Stranded Copper Wire	6	£6.00
CON-2356	Sch 40 Blk Pipe	3	£20.00
CON-2356	4x8x3/4 Cos Plywood	2	£10.00
CON-2356	3/4" EMT	2	£50.00
CON-2356	Duplex Ivy Rec	1	£100.00
CON-1234	Sch 40 Blk Pipe	1	£20.00
CON-1234	3/4" Locknut	2	£1.50

Activity 3: Third Normal Form

To place the data into Third Normal Form 3NF I need to ensure that all attributes are only dependent on the Primary Key, and not Non-Key Attributes. This can be achieved in two steps:

1. Test each attribute for dependency on the primary key.

2. Remove all transitive dependencies to a new entity with the correct primary key and relationship.

For this particular set of data, both the *Item* and *ItemOrder* entities are already in the Third Normal Form.

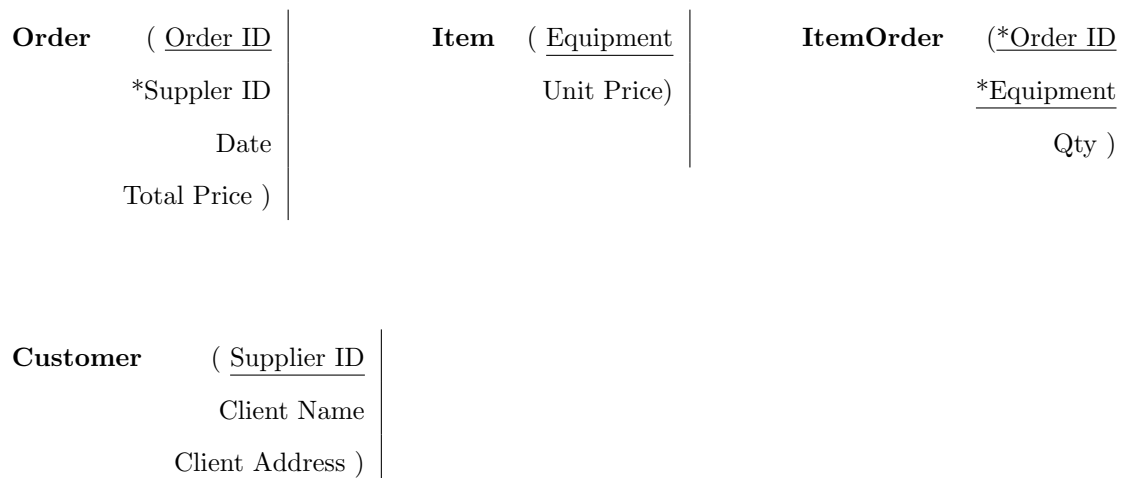### Step 1: Testing for Transitive Dependency

Order Entity

Primary Key	Attribute	Transitive Dependency?
Order ID	Supplier ID	Yes: Supplier ID can be found if we know Client Name or Address
Order ID	Client Name	Yes: Client Name can be found if we know Supplier ID or Address
Order ID	Client Address	Yes: Client Address can be found if we know Client Name or Supplier ID
Order ID	Date	No: Only dependent on Primary Key
Order ID	Total Cost	No: Only dependent on Primary Key

Using this table I have identified that *Supplier ID, Client Name,* and *Client Address* all have transitive dependencies, and need to be moved to a new entity.

Step 2: Entity, Relationship and Key

I created a new entity called *Customer*, and moved the three attributes with transitive dependencies into it. I then made *Supplier ID* the Primary Key and created a relationship between the *Customer* and *Order* entities by repeating the *Supplier ID* attribute as a foreign key in the *Order* entity.

Entities in Third Normal Form: Diagram

Order   (<u>Order ID</u>   **Item**   (<u>Equipment</u>   **ItemOrder**   (<u>*Order ID</u>

             *Suppler ID   Unit Price)   <u>*Equipment</u>

               Date   Qty)

          Total Price)

Customer   (<u>Supplier ID</u>

           Client Name

        Client Address)

Entities in Third Normal Form: With Data

Order entity

Order ID	Supplier ID	Date	Total Price
CON-1234	032	16/04/2015	23.00
CON-2237	168	14/12/2014	99.00
CON-2356	169	12/02/2015	280.00
CON-3664	527	16/01/2015	36.00

Item entity

Equipment	Unit Price
3/4" EMT	50.00
3/4" Locknut	1.50
4x8x3/4 Cos Plywood	10.00
Butterfly Valve	5.00
Duplex Ivy Rec	100.00
Sch 40 Blk Pipe	20.00
Thin Stranded Copper Wire	6.00

ItemOrder entity

Order ID	Equipment	Qty
CON-1234	3/4" Locknut	2
CON-1234	Sch 40 Blk Pipe	1
CON-2237	3/4" Locknut	6
CON-2237	Butterfly Valve	2
CON-2237	Sch 40 Blk Pipe	4
CON-2356	3/4" EMT	2
CON-2356	4x8x3/4 Cos Plywood	2
CON-2356	Duplex Ivy Rec	1
CON-2356	Sch 40 Blk Pipe	3
CON-3664	Thin Stranded Copper Wire	6

Customer entity

Supplier ID	Client Name	Client Address
032	Grand Design Ltd	Unit 12,Stoneleigh Park, CV8 2UV
168	Coventry Building Services Ltd	Units 2-4, Binley Industrial Estate, CV3 2WL
169	Ricoh Builds Ltd	34,Lythalls La Industrial Estate, CV6 6RG
527	Allied Construction Ltd	32-34, Bilton Industrial Estate, CV3 5YB

Activity 4: ER Diagram

Task 2: Database Development

For this task I will provide the SQL statement I used to complete each question activity, I will then explain each part of the SQL statement, and give screen-shots of before and after the statement was executed (if applicable).

Each SQL statement was written and tested using ORACLE 11g Express Edition and ORACLE Application Express.

Question 1

To begin I created the two tables which have no dependency on any other table: *Aircraft* and *Airline*

Aircraft

Listing 1: CREATE AIRCRAFT

```
1  CREATE TABLE Aircraft
2  (     aircraft_code VARCHAR2(5) PRIMARY KEY,
3        aircraft_type VARCHAR2(30) NOT NULL,
4        aircraft_price NUMBER(11,2) NOT NULL,
5  );
```

The statement begins with *'CREATE TABLE Aircraft'* which will create a table called *aircraft*. Inside the brackets the three fields, their data types, and any constraints are listed. Below is a table which will explain why I chose these data types and constraints for each field.

Identifier	Data-Type	Constraint	Explanation
aircraft_code	VARCHAR2(5)	PRIMARY KEY	All aircraft_codes start with 'C' and are followed by up to four numerical digits. This field will be used as the Primary Key of the table.
aircraft_type	VARCHAR(30)	NOT NULL	Contains a combination of alphanumeric characters of up to 30 characters in length. This field cannot be left empty.
aircraft_price	NUMBER(11,2)	NOT NULL	Stores large numeric values, with a precision of 11 and a scale of 2. This allows for prices up to 99 Thousand Million and two decimal places. This field cannot be left empty.

Airline

Listing 2: CREATE AIRLINE

```
1  CREATE TABLE Airline
2  (     airline_code CHAR(4) PRIMARY KEY,
3        airline_name VARCHAR2(20) NOT NULL,
4        airline_address VARCHAR2(60) NOT NULL,
5        airline_city VARCHAR2(15) NOT NULL,
6        airline_country VARCHAR2(15) NOT NULL
7  );
```

Identifier	Data-Type	Constraint	Explanation
airline_code	CHAR(4)	PRIMARY KEY	A combination of 4 alphanumeric characters. airline_code is also the Primary Key
airline_name	VARCHAR2(20)	NOT NULL	This field allows a variable length of characters, up to a maximum of 20. This field cannot be left empty.
airline_address	VARCHAR2(60)	NOT NULL	This field allows for a combination of alphanumeric characters up to a maximum length of 60. This field cannot be left empty.
airline_city	VARCHAR2(15)	NOT NULL	This field allows a combination of alphanumeric characters up to a maximum length of 15. This field cannot be left empty.
airline_country	VARCHAR2(15)	NOT NULL	This field allows for a combination of alphanumeric characters up to a maximum length of 15. This field cannot be left empty.

Purchase_Order

Listing 3: CREATE PURCHASE_ORDER

```
1  CREATE TABLE purchase_order
2  ( purchase_order_no NUMBER(3,0) PRIMARY KEY ,
3      airline_code CHAR(4) NOT NULL,
4      date_of_purchase DATE NOT NULL,
5      CONSTRAINT FK_1 FOREIGN KEY (airline_code) REFERENCES airline(airline_code)
6      );
```

Identifier	Data-Type	Constraint	Explanation
purchase_order_no	NUMBER(3,0)	PRIMARY KEY	This field allows for numeric input with a precision of 3 and a scale of 0, this allows values in the range of 001 to 999. This field also acts as the Primary Key.
airline_code	CHAR(4)	NOT NULL, FOREIGN KEY	A combination of 4 alphanumeric characters. This field is a foreign key; Referencing the *airline_code* field from the *Airline* table.
date_of_purchase	DATE	NOT NULL	

Ordered_Aircraft

Listing 4: CREATE ORDERED_AIRCRAFT

```
CREATE TABLE ordered_aircraft
(
purchase_order_no NUMBER(3,0) NOT NULL,
aircraft_code VARCHAR2(5) NOT NULL,
aircraft_quantity NUMBER(3,0) NOT NULL,
CONSTRAINT PK_ORDERCODE PRIMARY KEY(purchase_order_no, aircraft_code),
CONSTRAINT FK_PO FOREIGN KEY (purchase_order_no)
REFERENCES purchase_order(purchase_order_no);
CONSTRAINT FK_AC FOREIGN KEY (aircraft_code)
  REFERENCES aircraft(aircraft_code)
);
```

Identifier	Data-Type	Constraint	Explanation
purchase_order_no	NUMBER(3,0)	NOT NULL, PRIMARY KEY, FOREIGN KEY	This field allows for numeric input with a precision of 3 and a scale of 0. It is a foreign key referencing; *purchase_order_no* from *purchase_order*
aircraft_code	VARCHAR2(5)	NOT NULL, PRIMARY KEY, FOREIGN KEY	This field allows for alphanumeric input up to a maximum length of 5. It is part of a composite key and is also a foreign key referencing *aircraft_code* from *aircraft*.
aircraft_quantity.	NUMBER (3,0)	NOT NULL	This field allows for 3 digit numbers. This field cannot be left blank.

Question 2

Aircraft

Listing 5: INSERT INTO *AIRCRAFT*

```sql
INSERT ALL
INTO Aircraft VALUES ('C800' , 'CU-800 Commuter'        ,8000000000)
INTO Aircraft VALUES ('C8000', 'CU-8000 Stratocruiser',229000000)
INTO Aircraft VALUES ('C9000', 'CU-9000 Mesocruiser'  ,300000000)
INTO Aircraft VALUES ('C24'  , 'CU-24 Slipstream'      ,210000000)
INTO Aircraft VALUES ('C8'   , 'CU-8X Heavy Lifter'   ,100000000)
INTO Aircraft VALUES ('C10'  , 'CU-10 Exolinear'       ,400000000)
INTO Aircraft VALUES ('C900' , 'CU-900 Commuter'       ,190000000)
INTO Aircraft VALUES ('C80'  , 'CU-80 Cloud Hopper'   ,86000000)
INTO Aircraft VALUES ('C6'   , 'CU-6X Rapid Lifter'   ,280000000)
INTO Aircraft VALUES ('C22'  , 'CU-22 Executive Jet'  ,110000000)
INTO Aircraft VALUES ('C5'   , 'CU-X5 Spaceplane'      ,6000000000)
SELECT 1 FROM DUAL;
```

The SQL statement uses the following pattern:
INSERT ALL…INTO (table identifier) VALUES (data) .

This allows me to input multiple records into the table simultaneously. '*INTO Aircraft*' specifies that I am going to insert the following values into the *Aircraft* table.
The text in brackets following the word *VALUES* is the data which is input into that record.

'*SELECT 1 FROM DUAL*' is a requirement of INSERT ALL, so that it can function correctly.

EDIT	AIRCRAFT_CODE	AIRCRAFT_TYPE	AIRCRAFT_PRICE
📝	C800	CU-800 Commuter	80000000
📝	C8000	CU-8000 Stratocruiser	229000000
📝	C9000	CU-9000 Mesocruiser	300000000
📝	C24	CU-24 Slipstream	210000000
📝	C8	CU-8X Heavy Lifter	100000000
📝	C10	CU-10 Exolinear	400000000
📝	C900	CU-900 Commuter	190000000
📝	C80	CU-80 Cloud Hopper	86000000
📝	C6	CU-6X Rapid Lifter	280000000
📝	C22	CU-22 Executive Jet	110000000
📝	C5	CU-X5 Spaceplane	6000000000
			row(s) 1 - 11 of 11

Airline

Listing 6: INSERT INTO *AIRLINE*

```
INSERT ALL

INTO Airline VALUES ('BA07','British Airways PLC',
'Waterside, PO Box 365, Harmondsworth, UB7 0GB' ,'London', 'England')

INTO Airline VALUES ('UA09','United Airlines',
'77 W. Wacker Drive, Chigaco, IL 60601, United States', 'Chicago', 'United States')

INTO Airline VALUES ('AI06','Air India' ,
'Air-India Building, Nairman Point, Mumbai, 400 021' ,'Mumbai', 'India')

INTO Airline VALUES ('AC05','Air Coventry LTD' ,
'Coventry Univeristy, Priory Street, COventry, CV1 5FB' ,'Coventry', 'England')

INTO Airline VALUES ('RM04','Royal Mail Group',
'100 Victoria Embankment, London, EC4y 0HQ' ,'London', 'England')

INTO Airline VALUES ('IR01','Iran Air',
'1 Valiasr Street, Tehran' ,'Tehran', 'Iran')

SELECT 1 FROM DUAL;
```

EDIT	AIRLINE_CODE	AIRLINE_NAME	AIRLINE_ADDRESS	AIRLINE_CITY	AIRLINE_COUNTRY
✎	BA07	British Airways PLC	Waterside, PO Box 365, Harmondsworth, UB7 0GB	London	England
✎	UA09	United Airlines	77 W. Wacker Drive, Chigaco, IL 60601, United States	Chicago	United States
✎	AI06	Air India	Air-India Building, Nairman Point, Mumbai, 400 021	Mumbai	India
✎	AC05	Air Coventry LTD	Coventry Univeristy, Priory Street, COventry, CV1 5FB	Coventry	England
✎	RM04	Royal Mail Group	100 Victoria Embankment, London, EC4y 0HQ	London	England
✎	IR01	Iran Air	1 Valiasr Street, Tehran	Tehran	Iran
					row(s) 1 - 6 of 6

Purchase_Order

Listing 7: INSERT INTO *PURCHASE_ORDER*

```
INSERT ALL

INTO Purchase_Order VALUES (689,'BA07',TO_DATE ('28/04/2012', 'dd/mm/yyyy'))
INTO Purchase_Order VALUES (789,'UA09',TO_DATE ('24/06/2011', 'dd/mm/yyyy'))
INTO Purchase_Order VALUES (800,'AI06',TO_DATE ('30/07/2013', 'dd/mm/yyyy'))
INTO Purchase_Order VALUES (898,'AC05',TO_DATE ('06/06/2012', 'dd/mm/yyyy'))
INTO Purchase_Order VALUES (900,'RM04',TO_DATE ('03/11/2012', 'dd/mm/yyyy'))
INTO Purchase_Order VALUES (901,'IR01',TO_DATE ('03/11/2014', 'dd/mm/yyyy'))

SELECT 1 FROM DUAL;
```

EDIT	PURCHASE_ORDER_NO	AIRLINE_CODE	DATE_OF_PURCHASE
✎	689	BA07	04/28/2012
✎	789	UA09	06/24/2011
✎	800	AI06	07/30/2013
✎	898	AC05	06/06/2012
✎	900	RM04	11/03/2012
✎	901	IR01	11/03/2014
			row(s) 1 - 6 of 6

Ordered_Aircraft

Listing 8: INSERT INTO *ORDERED_AIRCRAFT*

```
1  INSERT ALL
2  INTO Ordered_Aircraft VALUES( 689, 'C800',14)
3  INTO Ordered_Aircraft VALUES( 689, 'C8000',8)
4  INTO Ordered_Aircraft VALUES( 689, 'C9000',5)
5  INTO Ordered_Aircraft VALUES( 689, 'C24',14)
6  INTO Ordered_Aircraft VALUES( 689, 'C8',7)
7  INTO Ordered_Aircraft VALUES( 789, 'C10',4)
8  INTO Ordered_Aircraft VALUES( 789, 'C900',30)
9  INTO Ordered_Aircraft VALUES( 789, 'C8000',8)
10 INTO Ordered_Aircraft VALUES( 800, 'C80',9)
11 INTO Ordered_Aircraft VALUES( 800, 'C6',11)
12 INTO Ordered_Aircraft VALUES( 800, 'C22',32)
13 INTO Ordered_Aircraft VALUES( 898, 'C5',6)
14 INTO Ordered_Aircraft VALUES( 898, 'C24',12)
15 INTO Ordered_Aircraft VALUES( 898, 'C800',25)
16 INTO Ordered_Aircraft VALUES( 898, 'C10',6)
17 INTO Ordered_Aircraft VALUES( 900, 'C8',8)
18 INTO Ordered_Aircraft VALUES( 900, 'C6',25)
19 INTO Ordered_Aircraft VALUES( 900, 'C80',8)
20 INTO Ordered_Aircraft VALUES( 901, 'C80',10)
21
22
23 SELECT 1 FROM DUAL;
```

EDIT	PURCHASE_ORDER_NO	AIRCRAFT_CODE	AIRCRAFT_QUANTITY
	689	C800	14
	689	C8000	8
	689	C9000	5
	689	C24	14
	689	C8	7
	789	C10	4
	789	C900	30
	789	C8000	8
	800	C80	9
	800	C6	11
	800	C22	32
	898	C5	6
	898	C24	12
	898	C800	25
	898	C10	6
		row(s) 1 - 15 of 19	

EDIT	PURCHASE_ORDER_NO	AIRCRAFT_CODE	AIRCRAFT_QUANTITY
📝	900	C8	8
📝	900	C6	25
📝	900	C80	8
📝	901	C80	10
			⊘ row(s) 16 - 19 of 19

Question 3

This question requires me to find the; Average, Minimum and Maximum price for all aircraft bought after '01/01/2012'.

Listing 9: AVG() MIN() MAX()

```
SELECT  AVG(ordered_aircraft.aircraft_quantity * aircraft_price) AS Average_Price,
MIN(ordered_aircraft.aircraft_quantity * aircraft_price) AS Minimum_Price,
MAX(ordered_aircraft.aircraft_quantity * aircraft_price) AS Maxium_Price
FROM ordered_aircraft JOIN aircraft
ON ordered_aircraft.aircraft_code = aircraft.aircraft_code
JOIN purchase_order
ON ordered_aircraft.purchase_order_no = purchase_order.purchase_order_no
WHERE purchase_order.date_of_purchase > '01/01/2012';
```

Firstly, to work out the price for all the aircraft bought, I multiply the quantity of aircraft ordered (*aircraft_quantity*) by the price of that particular aircraft (*aircraft_price*). To find the average, minimum and maximum of these prices then I need to use this calculation in conjunction with the *AVG()*, *MIN()* and *MAX()* SQL functions respectively. These functions will perform this calculation on every record in the specified tables and return the correct value. Once I selected the values, I used the *AS* keyword to label the output columns.
This can be seen in line 1 - 3 in the code above.
In total, this SQL statement uses three tables *ordered_aircraft* and *aircraft* for the calculations, and *purchase_order* for the conditional statement.
To be able to use the data from multiple tables, I need to use *JOINS*. *JOINS* allow me to combine common records from two or more tables by using the syntax structure;

JOIN table name ON table1.attribute = table2.attribute

For this problem I used two joins;

1. JOIN aircraft ON ordered_aircraft.aircraft_code = aircraft.aircraft_code

2. JOIN purchase_order ON ordered_aircraft.purchase_order_no = purchase_order.purchase_order_no

Join number One is used to select the correct relation attributes from both required tables.
Join number Two is used to ensure that the WHERE condition relates to the selected attributes.
This allowed me to select all the values I needed and also make sure that the value used in each calculation were from the same record.
Below is a screenshot of the statements' output.

AVERAGE_PRICE	MINIMUM_PRICE	MAXIUM_PRICE
4233375000	688000000	36000000000

Question 4

This question required me to: "Display the purchase order number, date, airline name, address, airline country for airlines where the total cost of an order is less than 10,000 million pounds. Your results should be in descending alphabetical order based on the airline code".

Listing 10: Question 4

```
SELECT
purchase_order.purchase_order_no,
purchase_order.date_of_purchase,
airline.airline_name,
airline.airline_address,
airline.airline_country,
ordered_aircraft.aircraft_quantity * aircraft.aircraft_price  AS total_cost
FROM airline
JOIN purchase_order
ON airline.airline_code=purchase_order.airline_code
JOIN ordered_aircraft
ON purchase_order.purchase_order_no=ordered_aircraft.purchase_order_no
JOIN aircraft
ON ordered_aircraft.aircraft_code=aircraft.aircraft_code
WHERE ordered_aircraft.aircraft_quantity * aircraft.aircraft_price < 10000000000
ORDER BY airline.airline_code DESC;
```

Lines 1 - 8 above I am using a SELECT ... FROM statement to select the required atributes from the tables. However not all of these attributes are from the same table, so I used multiple JOIN statements (lines 9 - 14) to combine the common attributes. On line 15 I used a WHERE keyword to specify the required condition (Total cost ¿ 10,000 Million pounds). Finally on Line 16, I sorted the results in descending order on *airine_code* by using ORDER BY ...DESC.

Below is a screenshot of the statements' output.

PURCHASE_ORDER_NO	DATE_OF_PURCHASE	AIRLINE_NAME	AIRLINE_ADDRESS	AIRLINE_COUNTRY	TOTAL_COST
789	06/24/2011	United Airlines	77 W. Wacker Drive, Chigaco, IL 60601, United States	United States	1600000000
789	06/24/2011	United Airlines	77 W. Wacker Drive, Chigaco, IL 60601, United States	United States	5700000000
789	06/24/2011	United Airlines	77 W. Wacker Drive, Chigaco, IL 60601, United States	United States	1832000000
900	11/03/2012	Royal Mail Group	100 Victoria Embankment, London, EC4y 0HQ	England	7000000000
900	11/03/2012	Royal Mail Group	100 Victoria Embankment, London, EC4y 0HQ	England	800000000
900	11/03/2012	Royal Mail Group	100 Victoria Embankment, London, EC4y 0HQ	England	688000000
901	11/03/2014	Iran Air	1 Valiasr Street, Tehran	Iran	860000000
689	04/28/2012	British Airways PLC	Waterside, PO Box 365, Harmondsworth, UB7 0GB	England	700000000
689	04/28/2012	British Airways PLC	Waterside, PO Box 365, Harmondsworth, UB7 0GB	England	2940000000
689	04/28/2012	British Airways PLC	Waterside, PO Box 365, Harmondsworth, UB7 0GB	England	1500000000
689	04/28/2012	British Airways PLC	Waterside, PO Box 365, Harmondsworth, UB7 0GB	England	1832000000
689	04/28/2012	British Airways PLC	Waterside, PO Box 365, Harmondsworth, UB7 0GB	England	1120000000
800	07/30/2013	Air India	Air-India Building, Nairman Point, Mumbai, 400 021	India	3520000000
800	07/30/2013	Air India	Air-India Building, Nairman Point, Mumbai, 400 021	India	774000000
800	07/30/2013	Air India	Air-India Building, Nairman Point, Mumbai, 400 021	India	3080000000
898	06/06/2012	Air Coventry LTD	Coventry Univeristy, Priory Street, COventry, CV1 5FB	England	2520000000
898	06/06/2012	Air Coventry LTD	Coventry Univeristy, Priory Street, COventry, CV1 5FB	England	2000000000
898	06/06/2012	Air Coventry LTD	Coventry Univeristy, Priory Street, COventry, CV1 5FB	England	2400000000

Question 5

This question required me to: "Display how many aircraft were ordered by British Airways in total and by each different airplane type."

Listing 11: Question5

```
1  SELECT ordered_aircraft.aircraft_quantity AS Total_aircraft_ordered,
2  aircraft.aircraft_type
3  FROM ordered_aircraft
4  JOIN aircraft
5  ON ordered_aircraft.aircraft_code=aircraft.aircraft_code
6  JOIN purchase_order
7  ON ordered_aircraft.purchase_order_no=purchase_order.purchase_order_no
8  JOIN airline
9  ON purchase_order.airline_code=airline.airline_code
10 WHERE airline.airline_code ='BA07';
```

Lines 1 - 3 use SELECT ... FROM to select the two rows needed for this task.

Lines 4 - 9 contains the joins required to combine the related data in the tables.

Lines 10 uses WHERE to define a condition. In this particular case it only selects data where airline_code is equal to $BA07$, which is the airline code for *"British airways PLC"*.

Below is a screenshot of the statements' output.

TOTAL_AIRCRAFT_ORDERED	AIRCRAFT_TYPE
14	CU-24 Slipstream
7	CU-8X Heavy Lifter
14	CU-800 Commuter
8	CU-8000 Stratocruiser
5	CU-9000 Mesocruiser

Question 6

This question required me to: "Produce a list of all the orders from all the airlines. The list should show the airline code, followed by the order number, followed by the code of the aircraft ordered, followed by the quantity of aircraft ordered and then the total cost of that order. The list should be arranged by airline code in descending order."

Listing 12: Question 6

```sql
SELECT
airline.airline_code,
ordered_aircraft.purchase_order_no,
aircraft.aircraft_code,
ordered_aircraft.aircraft_quantity,
(ordered_aircraft.aircraft_quantity * aircraft.aircraft_price) AS total_price
FROM airline
JOIN purchase_order
ON purchase_order.airline_code=airline.airline_code
JOIN ordered_aircraft
ON purchase_order.purchase_order_no=ordered_aircraft.purchase_order_no
JOIN aircraft
ON ordered_aircraft.aircraft_code=aircraft.aircraft_code
ORDER BY airline.airline_code DESC;
```

Lines 1 - 7 use SELECT ... FROM to select the five rows needed for this task and specify their location.
Lines 8 - 13 join the tables so to combined the related data to achieve contiguous output.
Line 14 ORDER BY ...DESC to sort the data in descending order via the *airline_code* column.
Below is a screenshot of the statements' output.

AIRLINE_CODE	PURCHASE_ORDER_NO	AIRCRAFT_CODE	AIRCRAFT_QUANTITY	TOTAL_PRICE
UA09	789	C900	30	5700000000
UA09	789	C10	4	1600000000
UA09	789	C8000	8	1832000000
RM04	900	C6	25	7000000000
RM04	900	C80	8	688000000
RM04	900	C8	8	800000000
IR01	901	C80	10	860000000
BA07	689	C24	14	2940000000
BA07	689	C800	14	1120000000
BA07	689	C8	7	700000000
BA07	689	C8000	8	1832000000
BA07	689	C9000	5	1500000000
AI06	800	C22	32	3520000000
AI06	800	C6	11	3080000000
AI06	800	C80	9	774000000
AC05	898	C800	25	2000000000
AC05	898	C5	6	36000000000
AC05	898	C24	12	2520000000
AC05	898	C10	6	2400000000

Question 7

This question required me to : ". Display the order details, airline details and the aircraft details where more than 10 aircraft of a specific type were ordered."

Listing 13: Question 7

```
1   SELECT purchase_order.*,
2   ordered_aircraft.aircraft_quantity,
3   airline.airline_name,
4   airline.airline_address,
5   airline.airline_city,
6   airline.airline_country,
7   ordered_aircraft.aircraft_code,
8   aircraft.aircraft_type,
9   aircraft.aircraft_price
10  FROM purchase_order
11  JOIN airline
12  ON airline.airline_code=purchase_order.airline_code
13  JOIN ordered_aircraft
14  ON purchase_order.purchase_order_no=ordered_aircraft.purchase_order_no
15  JOIN aircraft
16  ON ordered_aircraft.aircraft_code=aircraft.aircraft_code
17  WHERE ordered_aircraft.aircraft_quantity > 10;
```

Lines 1 - 10 select the required rows from the data tables.

Lines 11 - 16 Joins the tables on the common data elements.

Line 17 is a condition statement, where the *aircraft_quantity* is greater than ten.

Below is a screenshot of the statements' output.

PURCHASE_ORDER_NO	AIRLINE_CODE	DATE_OF_PURCHASE	AIRCRAFT_QUANTITY	AIRLINE_NAME	AIRLINE_ADDRESS	AIRLINE_CITY	AIRLINE_COUNTRY	AIRCRAFT_CODE	AIRCRAFT_TYPE	AIRCRAFT_PRICE
898	AC05	06/06/2012	25	Air Coventry LTD	Coventry Univeristy, Priory Street, COventry, CV1 5FB	Coventry	England	C800	CU-800 Commuter	80000000
689	BA07	04/28/2012	14	British Airways PLC	Waterside, PO Box 365, Harmondsworth, UB7 0GB	London	England	C800	CU-800 Commuter	80000000
898	AC05	06/06/2012	12	Air Coventry LTD	Coventry Univeristy, Priory Street, COventry, CV1 5FB	Coventry	England	C24	CU-24 Slipstream	210000000
689	BA07	04/28/2012	14	British Airways PLC	Waterside, PO Box 365, Harmondsworth, UB7 0GB	London	England	C24	CU-24 Slipstream	210000000
789	UA09	06/24/2011	30	United Airlines	77 W. Wacker Drive, Chigaco, IL 60601, United States	Chicago	United States	C900	CU-900 Commuter	190000000
900	RM04	11/03/2012	25	Royal Mail Group	100 Victoria Embankment, London, EC4y 0HQ	London	England	C6	CU-6X Rapid Lifter	280000000
800	AI06	07/30/2013	11	Air India	Air-India Building, Nairman Point, Mumbai, 400 021	Mumbai	India	C6	CU-6X Rapid Lifter	280000000
800	AI06	07/30/2013	32	Air India	Air-India Building, Nairman Point, Mumbai, 400 021	Mumbai	India	C22	CU-22 Executive Jet	110000000

Question 8

This question required me to: "We have discovered an error in our spreadsheet. The C800 aircraft should have been 100 million not 80 million please update the database based on this change."

Listing 14: Question 8

```
1  UPDATE aircraft
2  SET aircraft_price=100000000
3  WHERE aircraft_code='C800'
```

Below is a screen shot of the updated record.

EDIT	AIRCRAFT_CODE	AIRCRAFT_TYPE	AIRCRAFT_PRICE
✎	C800	CU-800 Commuter	100000000

Question 9

This question required me to: ". Air Coventry LTD has changed its name to Coventry University Airways please update the information."

Listing 15: Question 9

```
1  UPDATE airline
2  SET airline_name='Coventry University Airways'
3  WHERE airline_name='Air Coventry LTD'
```

Below is a screen shot of the updated record.

✎	AC05	Coventry University Airways	Coventry Univeristy, Priory Street, COventry, CV1 5FB	Coventry	England

Task 3 : Poster of Ethics

Task 4 : Amazon recommendation system

What is a Recommendation System?

Recommendation systems are commonly described as:

"A way of presenting information on items and products that are likely to be of interest to the reader."
[1]

Recommendation systems work via the use of Data Mining. Data Mining is a process where data is gathered and analysed to find specific patterns. These patterns can provide meaningful and insightful data which can then be relayed to the consumer.

The data gathered for a recommendation system comes in many varieties and is often tailored to the recommendation systems' main purpose. For Example, a system may collect data on a users' Hobbies, Interests, Opinions, Habits, Recent search history, Recent Purchase History and many more areas.

Data Gathering and Preparation

Amazon is an incredibly popular E-Commerce website which caters to millions of customers everyday. This is partially due to its intricate and extensive data mining operations which result in a very effective recommendation system. The effectiveness of this system can be seen by looking at their percentage sales increases;

"The company reported a 29% sales increase to \$12.83 billion during its second fiscal quarter, up from \$9.9 billion during the same time last year."[2]

Data

To make the recommendation system as effective as possible, Amazon must collect a huge amount of data on each customer and more importantly they must collect the right data. In this section I will explore what kind of data this recommendation system should collect.
At Amazon, the recommendation system is used to personalize the shopping environment for each customer, so the store changes drastically depending on the customer.

Completed Purchases
> Products the customer has spent money on.

Incomplete purchases
> Abandoned shopping carts or specific items dropped from the cart.

Personal item ratings/reviews
> Ratings will change the recommendations depending on the quality of the rating.

Demographic information
> Such as Location, Age, Profession, Type of education etc... will offer different recommendations depending on which demographic criteria the customer meets.

Number of views
> How many times was thaty particular item, or similar items viewed before purchase.

Time spent before purchase
How long it takes for a customer to purchase an item

Click-Through Rate
How often a customer clicks on a recommendation.

Conversion rate
How often the customer purchases an item that has been recommended.

Bibliography

References for Task 3: A Poster of Ethics

References for Task 4: A Recommendation system

1. http://www.webopedia.com/TERM/R/recommender_systems.

2. http://fortune.com/2012/07/30/amazons-recommendation-secret/