CS990 Database Fundamentals.

Classwork 2.

THIS IS AN INDIVIDUAL TASK AND MUST BE ACCOMPLISHED WITHOUT COLLABORATION, COLLUSION OR THE SHARING OF SOLUTIONS. FAILURE TO FOLLOW THIS INSTRUCTION WILL RESULT IN DISCIPLINARY ACTION BEING TAKEN.

1. Aim Of The Assignment:
Your task during the classwork is to design and construct a database and use it to store and retrieve data.

2. Task:
Read the following description of a data model. From the specification, produce:

1. a logical design for the database
2. the SQL code to construct and populate the tables with a minimal amount of data that is needed to carry out the queries specified.
3. The SQL code to answer the queries listed below along with an explanation of the queries.
4. A listing of the output of each of the queries.
5. A critique of your design and implementation.

The database must be built using Oracle.

A database is required to store information concerning a manufacturing company as follows:

A database is to be used by an organisation to store data about orders. Each order is identified by an order number. Each order has an order date and contains a number of order lines. Each order line specifies a part number and a quantity for the part. Parts are either manufactured by the organisation or 'bought in'. In the case of 'bought in' parts it is important to store information about the cost of the part. Parts that are ‘bought in’ are obtained from suppliers whose names and addresses are stored in the database. In addition, each supplier has a bank account with a branch code and account number. It is possible that some parts held by the organisation have not been ordered. A supplier can supply many parts and the same part can be sourced from a number of suppliers. Some parts are composites; i.e. they are assemblies of other parts. A single part may be used in a number of different composite parts. Manufactured parts may be assembled to order or may be made in bulk and then held in stock. For stock parts, it is important that the location is stored. Parts that are assembled to order require a lead time to be recorded.

An enhanced entity relationship model was developed to represent the database and included the following entities, attributes and relationships:

Entity meaning

Part: The parts that are stored
Manufactured: Manufactured parts
Bulk part: parts manufactured in bulk
To order: Parts made to order
Bought: Parts that are bought in
Used in: Linker entity to show parts breakdown
Order: Orders placed for parts
Order line: The lines on each order
Supplier part: Linker entity used to connect bought in parts with suppliers
Bank account: Bank accounts belonging to suppliers

List of entities and their attributes

Part (PID, Description)
    Manufactured Part (PID)
        Bulk part (PID, Location)
        To order (PID, Lead time)
    Bought (PID, Cost)
Used in (UID, Quantity)
Order (OID, Order date)
Order line (OLID, Quant)
Supplier (SID, Name, Address)
Supplier part (SPID, Price)
Bank account (BAID, Sort code, Account number)

Relationships

Relationship name	Entities	Degree	Optionality
Ool	Order, Order line	1:N	Obligatory on both
Pol	Part, Order line	1:N	Optional on Part Obligatory on Order line
Pu1	Part, Used in	1:N	Optional on Part Obligatory on Used in
Pu2	Part, Used in	1:N	Optional on Part Obligatory on Used in
Bp	Bought Supplier part	1:N	Obligatory on both
Ss	Supplier Supplier part	1:N	Obligatory on both
Sa	Supplier, Bank account	1:1	Obligatory on both

Assumptions:

All suppliers supply at least one part.

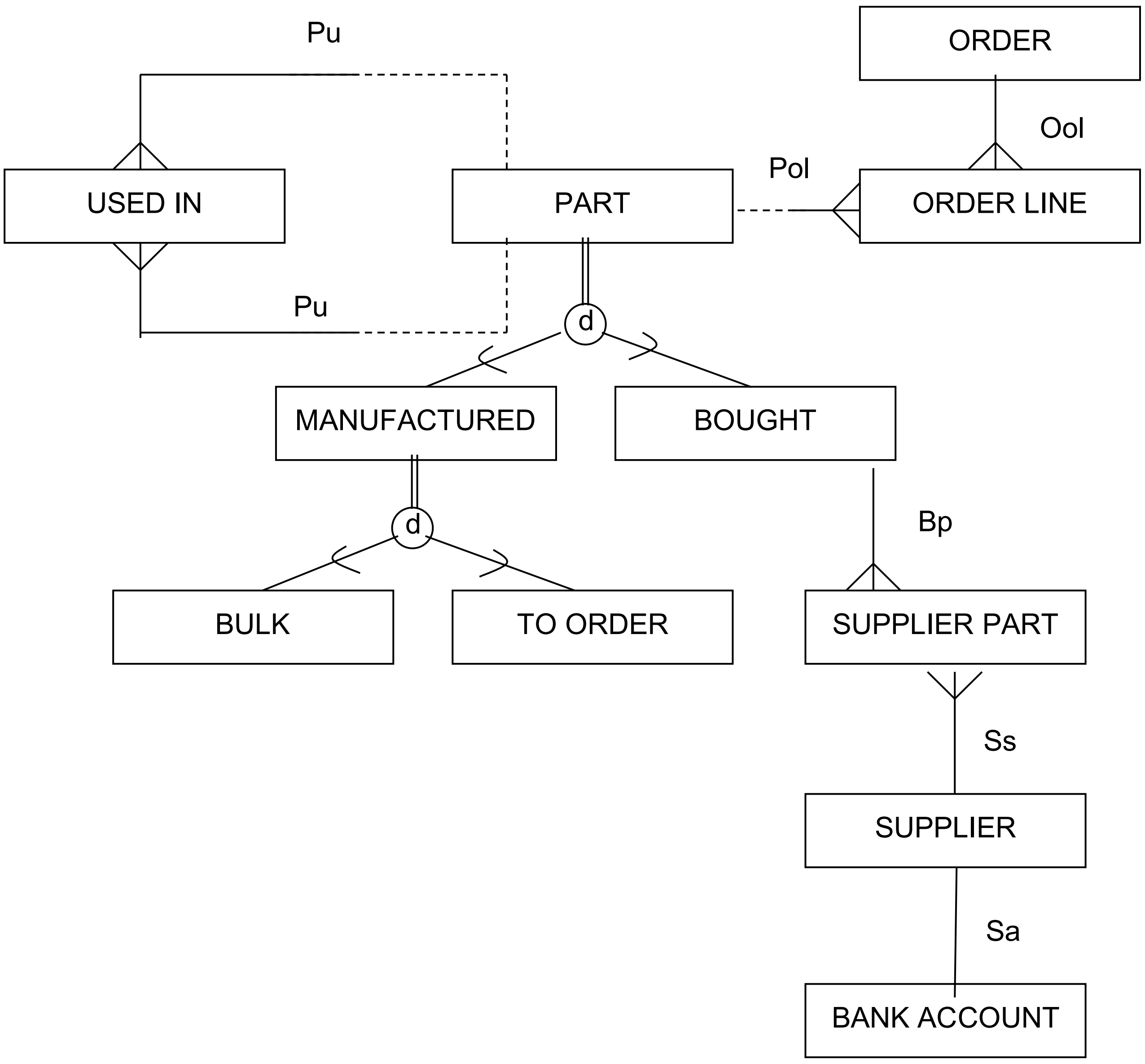

Figure 1. Enhanced Entity Relationship model diagram.

3. Logical design

Table structures should be written down in the following format:
TABLE_NAME(<u>Primary-key-attribute</u>,Non-key-attribute1,Non-key-attribute2.....).

Using the enhanced entity relationship model write down a table structures for each entity taking care that:

- each attribute becomes a column.
- the unique identifier becomes the primary key and is indicated by underlining
- subtype/supertype entities are represented in one of three methods described in the lectures

Use Table CW-1 as a guide to the way of representing the relationships between entities. Write down table structures or modify existing structures to represent relationships in the system.

	1 : 1	1 : N	N : M
Obligatory on neither	New table to represent relationship Post identifiers as candidate keys	New table to represent relationship Post identifiers as candidate keys	New table to represent relationship Post identifiers as candidate keys
Obligatory on one	Post identifier of non-obligatory to obligatory table	New table to represent relationship Post identifiers as candidate keys	-
Obligatory on many	-	Post identifier of "one" table to "many" table	New table to represent relationship Post identifiers as candidate keys
Obligatory on both	Post all attribute into one table	Post identifier of "one" table to "many" table	New table to represent relationship Post identifiers as candidate keys

Table CW-1: Representing relationships in tables.

4 Creating and loading the database
Use appropriate integrity constraints. Populate each table with a limited set of data i.e. only enough to show that the queries work.

5 Querying the database
You now need to write some queries on your database. The queries must be useful queries and not artificially constructed simply to fulfil the criteria listed. All queries require a WHERE clause of the form '…WHERE ATTRIBUTE = Value…' to limit the rows returned (Value can be a text, numeric or date value). Write
SQL statements that will
(i) carry out a join between two tables and use the group by clause.
(ii) execute a sub-query.
(iii) execute a correlated-query.
(iv) carry out a self join that uses primary key/foreign key attributes.

The output of Oracle SQL queries can be captured in a file by typing:

```
spool outfile
```

at the SQL prompt. All screen output is then copied to a file with the name outfile.lst. The spooling can be stopped by typing:

```
spool off
```

at the SQL prompt.

6. Submission
Your submission should consist of the following.

1. A list of the table structures produced by logical design showing the attributes and primary keys. (15 %)
2. The SQL create statements (including the specification of integrity constraints) for creating the tables. (25%)
3. The SQL insert statements for populating the tables with a small sample of data. (15%)
4. The SQL queries together with a narrative explanation of each query (do not paraphrase the SQL commands) and its output. (25%)
5. A 200 word critique of your database, highlighting the strengths and weaknesses of your solution and giving reasons for decisions that you have taken in the design and implementation. (20%)

Items (1) to (5) should be combined into a single document in the order shown and submitted through the link on the class Myplace page. The document must not be zipped and must not contain screen shots. All work will be evaluated for originality. Submission must be completed by 12.00noon on Monday 4th March 2019.

This exercise is worth 25% of the overall marks for this module