1. What are the possible advantages of using distributive processing in a database system environment? [06]

The possible advantages of using distributive processing in a DBS environment are as follows:

a. Reduction of a dependence on a central system should positively affect throughout & hence productivity.
b. Convenience: it might not be practical or productive to depend on a central database system.
c. Distributive processing can ensure that the best resources are brought together and utilized in a system.
d. Autonomy (independence) of processing (consequence of (a)).
e. Easy and fast communication (over long distances).
f. Readily tailored to the structure of the organization; easily scaled in size.
g. Improved processing speed & performance since local application programs may benefit from having information closer than would be the case of a central system (consequence of (a)).
h. More easily designed to facilitate multiple users.
i. Improves reliability: if one machine fails, the whole system does not fail.

2. What are the problems that can be experienced from having non-normalized files in a DBS? [08]

The problems that can be experienced from having un-normalized files in a system include the following:

a. Modification anomalies which include:
   - ✓Insertion anomaly: Can't insert data when it is desirable; have to wait on some future data, due to organization of the data structure.
   - ✓Deletion anomaly: Deletion of some undesirable aspect(s) of data necessarily means deletion of some other desirable aspect(s) of data.
   - ✓Update anomaly: Update of some aspect(s) of data necessarily means update of other aspect(s) of data.
b. Data redundancy that leads to the anomalies mentioned above.
c. Inefficient file manipulation; lack of ease of data access.
d. Un-normalized files in a system tend to reduce (or seriously impair) the likelihood of achieving logical data independence.
e. Compromise on the integrity of data.
f. Pressure on programming effort to make up for poor design.

3. Relation R(A B C . . . K) is in 1NF only. The following FDs hold:
   a) Primary key is (ABC)
   b) A –> D
   c) B –> E
   d) F –> G,H,I
   e) J –> B
   f) K –> C
   g) (AF) –>> J/K

   By repeatedly using Heath's or Fagin's theorems, decompose R into a set of 5NF relations. Show how the new relations will be keyed. [15]

Solution to Q3

Relation R $\{\underline{A, B, C, D}, E, F, G, H, I, J, K\}$ is in 1NF.

FDs are as follows:
FD1: Primary key is (ABC)
FD2: A → D
FD3: B → E
FD4: F → G, H, I
FD5: J → B
FD6: K → C
FD7: (AF) –>> J/K

Note: Throughout the solution, primary key attributes are underlined.

Step 2: Obtain 2NF Relations

R is not in 2NF due to FD2 and FD3. Using Heath's theorem, R is decomposed to

$R_1 \{\underline{A,} D\}$ $R_2 \{\underline{B,} E\}$ $R_3 \{\underline{A, B, C,} F, G, H, I, J, K\}$

Step 3: Obtain 3NF Relations

R_1 & R_2 are in 3NF but R_3 is not due to FD4. Using Heath's theorem, R_3 is decomposed to

$R_4 \{\underline{F,} G, H, I\}$ $R_5 \{\underline{A, B, C,} F, J, K\}$

Step 4: Obtain BCNF Relations

R_4 is in BCNF but R_5 is not since FD5 and FD6 are in play. Using Heath's theorem, R_5 is decomposed to

$R_6 \{\underline{J,} B\}$ $R_7 \{\underline{K,} C\}$ $R_8 \{\underline{A,} F, J, K\}$

At this point, we are not sure how to key R_8 since we were not told that A –> (F, J, K). However, the MVD of FD7 is instructive.

Step 4: Obtain 4NF Relations

Relations R_6, R_7 are in 4NF. Relations R_1, R_2, R_4 are also in 4NF. Relation R_8 is not in 4NF due to the MVD of FD7 as described above. Using Fagin's theorem, R_8 is decomposed to

$R_9 \{\underline{A, F, J}\}$ $R_{10} \{\underline{A, F, K}\}$

Step 5: Obtain 5NF Relations

The 5NF relations derived from the case are:
$R_1\{\underline{A}, D\}$ $R_2\{\underline{B,} E\}$ $R_4\{\underline{F}, G, H, I\}$ $R_6\{\underline{J,} B\}$ $R_7\{\underline{K,} C\}$ $R_9\{\underline{A, F, J}\}$ $R_{10} \{\underline{A, F, K}\}$

4. The following atomic data elements were taken from the data dictionary of an inventory management system:

Item Number [ITM#]	Item Name [ITMNAME]
Item Last Price [ITMLP]	Item On Hand Quantity [ITMQOH]
Item Average Price [ITMAP]	Item Category Code [ITMCTG]
Category Description [CTGDES]	Item Supplier [ITMSPLR]
Order# [ORD#]	Order Date [ORDDATE]
Item Ordered [ORDITM]	Quantity Ordered [ORDQTY]
Order Status [ORDSTS]	Invoice Number [INV#]
Receipt Date [RCVDATE]	Item Received [RCVITM#]
Quantity Received [RCVQTY]	Invoice's Related Order [INVORD#]
Requisition Date [RQSDATE]	Requisition Number [RQS#]
Department Requesting [RQSDEPT]	Issuance Date [ISSDATE]
Requisition Honor Date [RQSHDATE]	Quantity Issued [ISSQTY]
Issuance Number [ISS#]	Item Issued [ISSITM]
Item Issue Price [ISSPRC]	Department Receiving the Issue [ISSDPT]

Group the attributes into related entities. Then for each entity, identify the FDs, and then normalize it. Grade will be assigned as follows:

4.1 Identification of all possible FDs. [11]

4.2 Putting the elements into normalized relations. You may introduce elements as required with appropriate explanation. [44]

For this question, there are two possible approaches:

- Start with a large 1NF relation including all the attributes listed. Identify FDs and systematically decompose.
- By observation, identify obviously related attributes and group them together into relations (entities). Identify FDs and systematically decompose.

The answer to this question uses the second approach. Entities identified are as follows:

4A. **Inventory** {Itm#, ItmName, ItmQoh, ItmCtg, Itmap, CtgDes, ItmSplr, Splrnm}

The FDs are:
FD1: Itm# → ItmName, ItmQoh, ItmCtgCode, Itmap, CtgDes, ItmSplrCode, SplrName
FD2: ItmCtg → CtgDes
FD3: ItmSplr → SplrName

Decompose by Heath's theorem to obtain

Item {Itm#, ItmName, ItmQOH, ItmCtgCode, ItmAP, ItmLP} PK [Itm#]
Category {CtgCode, CtgDes} PK[CtgCode]
Supplier {SplrCode, SplrName} PK [SplrCode]

Note: ItmSplrCode is not included in entity **Item** as a foreign key since what actually exists between **Item** & **Supplier** is a M:M relationship. A new relation will be required to implement this:

ItemSupplierMatrix {Itm#, ItmSplrCode} PK [Itm#, ItmSplrCode]

4B. **Purchase Orders** {Ord#, OrdItm#, OrdQty, OrdSts, OrdDate, OrdSplrCode}

If we assume that purchase order numbers are repeatable over time, the following are the FDs:
FD4: [Ord# OrdSplrCode, OrdDate] → OrdSts
FD5: [Ord#, OrdSplrCode, OrdDate, OrdItm#] → OrdQty

If we assume that purchase order numbers are not repeatable over time, the following are the FDs:
FD4b: Ord# → OrdSplrCode, OrdDate, OrdSts
FD5b: [Ord#, OrdItm#] → OrdQty

We are not sure which situation will prevail, so it is prudent to cater for either. To do this, we decompose by Heath's theorem, and introduce a surrogate to obtain

OrderSummary {Ord#, OrdSplrCode, OrdSts, OrdDate, OrderRef#} PK [OrdRef#]
OrderDetail {OrdRef#, OrdItm#, OrdQty} PK [OrdRef#, OrdItm#]

4C. **PurchasedGoods** {Inv#, RcvDate, InvOrd#, InvTotal, InvSplrCode, InvDate, RcvItm, RcvQty, RcvPrc}

Similar to 4B, it is not prudent to assume that invoice numbers coming from various suppliers will be unique. With this in mind, we tentatively construct the following FDs:
FD6: [Inv#, InvSplrCode, InvDate, InvOrd#] → RcvDate, InvTotal
FD7: [Inv#, InvSplrCode, InvDate, InvOrd#, RcvItm#] → RcvQty, RcvPrc

As above, we play safe and decompose by Heath's theorem, and introduce a surrogate to obtain

InvoiceSummary {Inv#, InvSplrCode, RcvDate, InvOrdRef#, InvTotal, InvDate, InvRef#} PK [InvRef#]
InvoiceDetail {InvRef# , RcvItm#, RcvQty, RcvPrc} PK [InvRef#, RcvItm#]

Note: New attributes are **InvDate, RcvPrc, and InvTotal**. Also note that with the introduction of **InvOrdRef#** in the **InvoiceSummary** instead of **InvOrd#**, we do not need to store **InvSplrCode**, since the supplier can be deduced by simply tracking the related purchase order. However, this is a good example of having a little duplication for increased convenience.

4D. **Requisition** {Rqs#, RqsDate, RqsDept#, RqsHDate, RqsItm#, RqsQty}

This is going to be similar to 4B and 4C with one exception: since the requisitions are internal to the organization, we can control the requisition number to make sure that it is unique. With this in mind, the FDs are:
FD8: Rqs# → RqsDate, RqsDept#, RqsHDate
FD9: [Rqs#, RqsItm#] → RqsQty

Decompose by Heath's theorem to obtain:

RequisitionSummary {Rqs# , RqsDate, RqsDept, RqsHDate} PK [Rqs#]
RequisitionDetail {Rqs#, RqsItm#, RqsQty} PK[Rqs# , RqsItm#]

4E. **InventoryIssued** {Iss#, IssDate, IssDept#, IssItm#, IssQty, IssPrc}

Similar to requisitions, issuance of inventory is completely internal; thus the Iss# can be designed to be unique. With this in mind, the FDs are:
FD10: Iss# → IssDate, IssDept, IssRqs# (note that **IssRqs#** is the related requisition for the issuance)
FD11: [Iss#, IssItm#] → IssQty, IssPrc

Decompose by Heath's theorem to obtain:

IssueSummary {Iss#, IssDate, IssDept, IssRqs#} PK [Iss#]
IssueDetail {Iss#, IssItm#, IssQty, IssPrc} PK [Iss#, IssItm#]

4F. **Department** {Dept#, DeptName} PK[Dept#]

This relation was implied in 4D and 4E: requisitions are made by departments, and issuance of inventory are made to departments, in response to the requisitions.

4G. **Highlights of the Case**

1. New attributes that were introduced:
   SplrCode – Supplier Code
   SplrName – Supplier name
   CtgCode – Category Code
   InvDate – Invoice date
   RqsItm# – Requisition item code
   RqsQty – Requisition quantity
   RcvPrc – Received item price (per item)
   IssRqs# – Issue's Related Requisition Number
   InvTotal – Invoice total
   Dept# – Department code
   DeptName – Department name

2. All relations are in 4NF.

3. Except for a few deliberate deviations, the principle of using unique attribute-names for the database was followed.

4.3 For extra credit, construct an ERD representing the case with its normalized relations. [30]