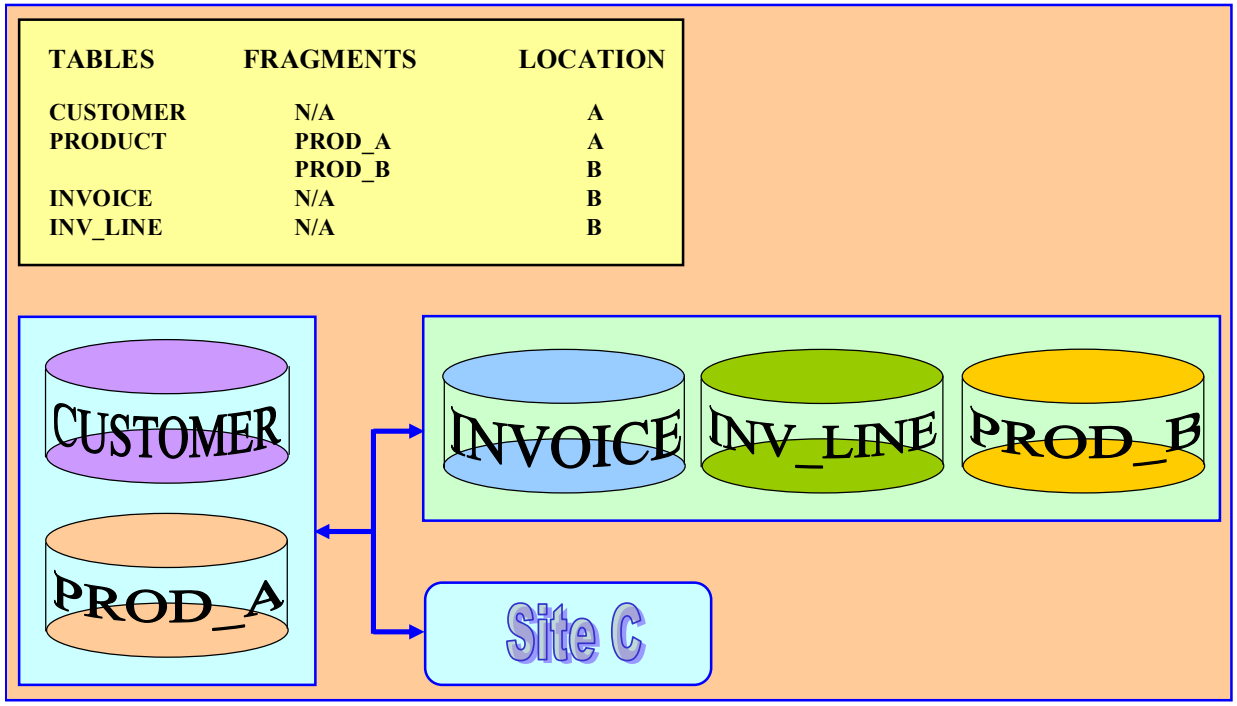
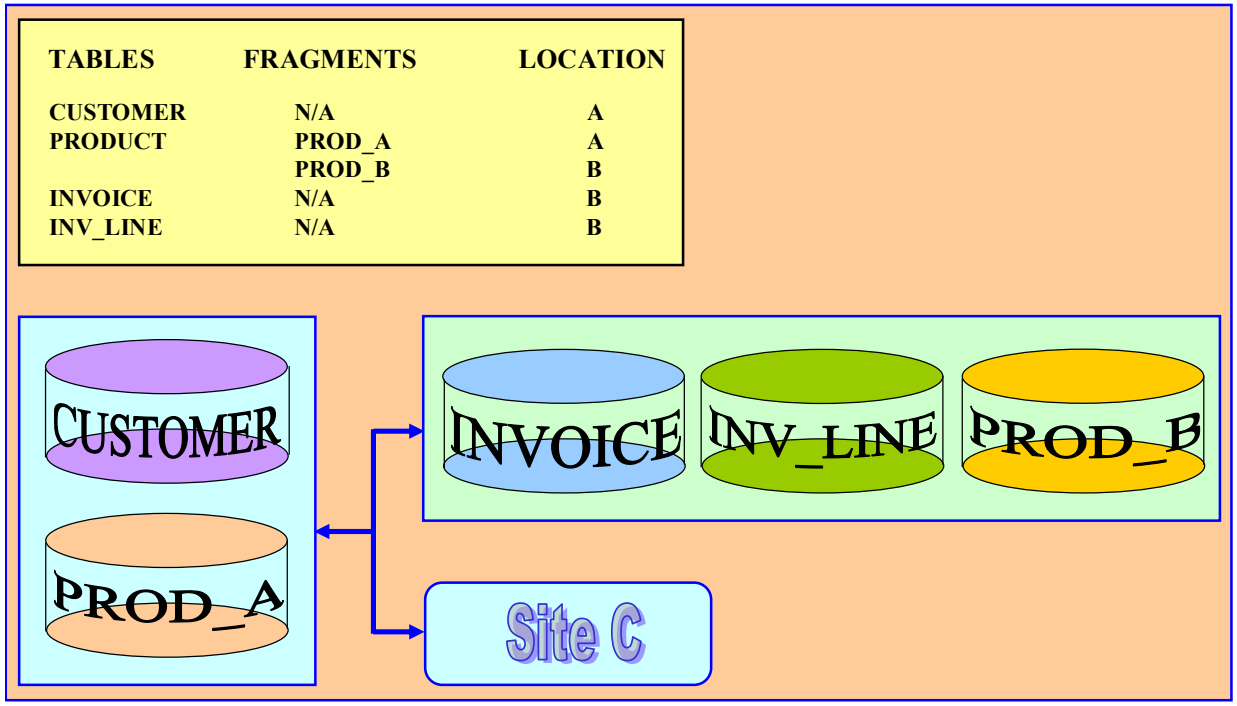
Ch12 Exercises Distributed DBMS

As usual, please ignore problem numbers and other oddly bulleted items ☺

1. **Specify the minimum types of operations the database must support to perform the following operations. These opertaions should include remote request, remote transaction, distributed transaction, and distributed requests in order to perform the following operations.**

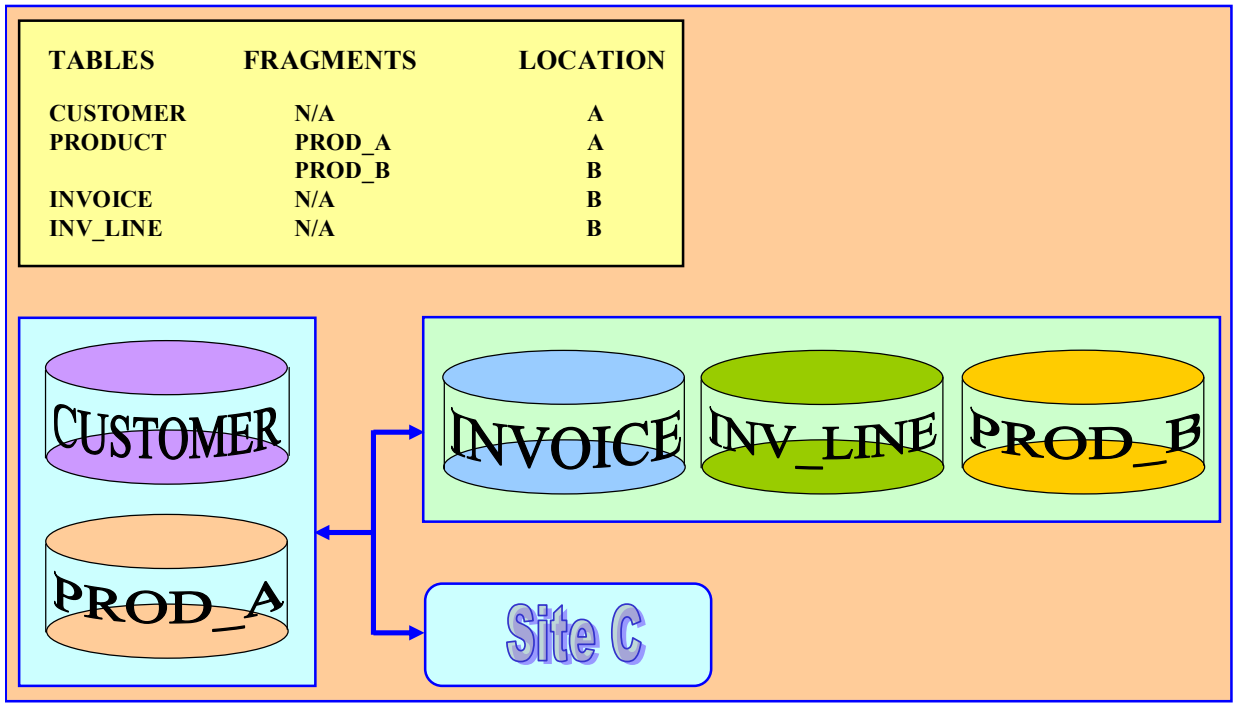


. SELECT *
FROM CUSTOMER;

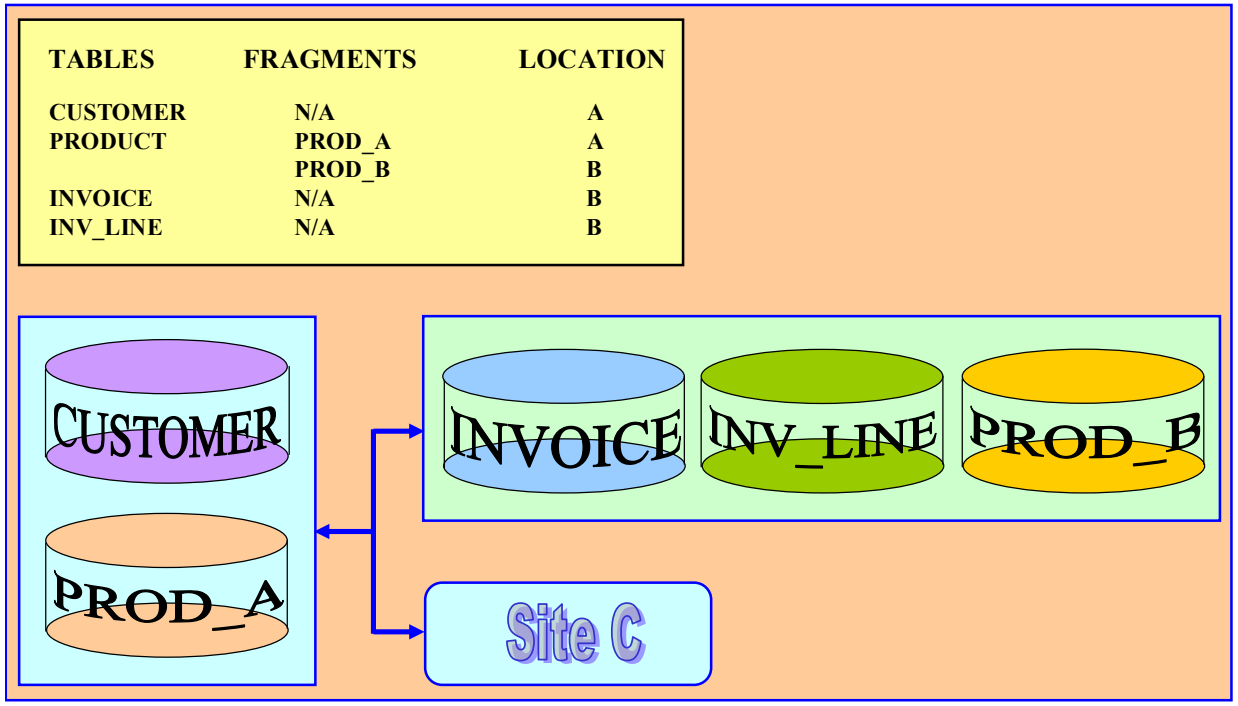


```
SELECT      *
  FROM      CUSTOMER;
```

This SQL sequence represents a *remote request*.

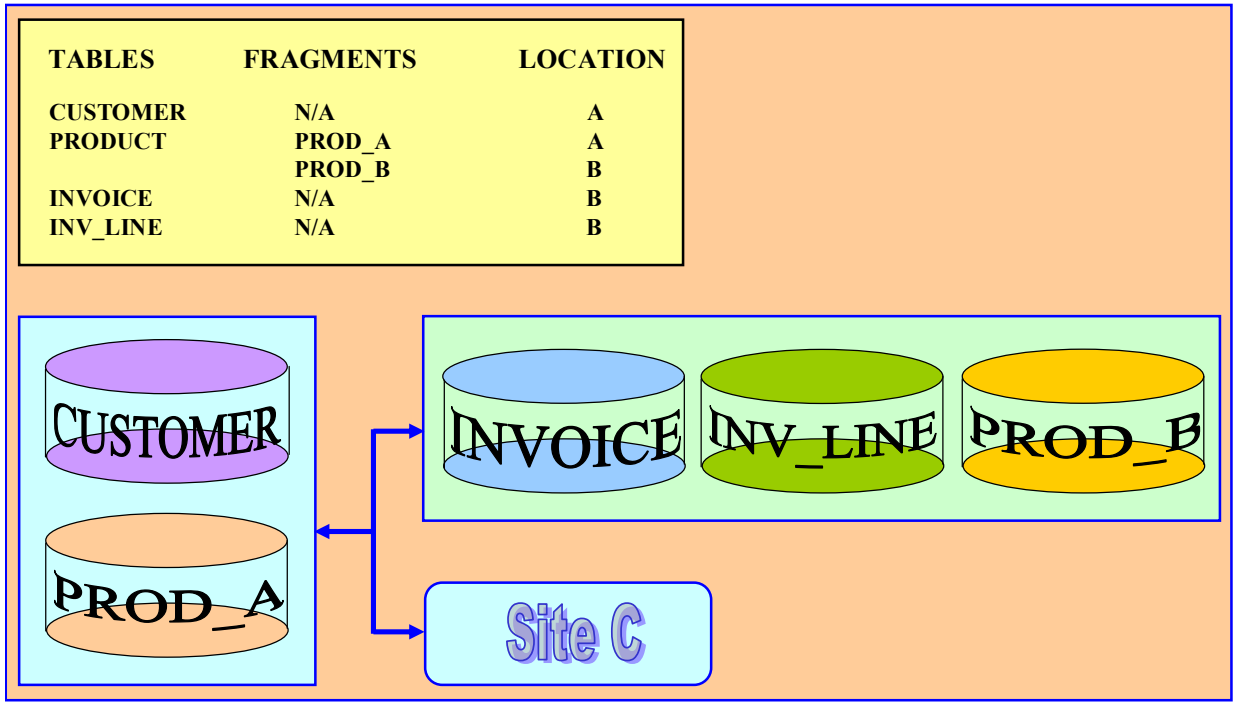


```
SELECT      *
   FROM     INVOICE
   WHERE    INV_TOTAL < 1000;
```

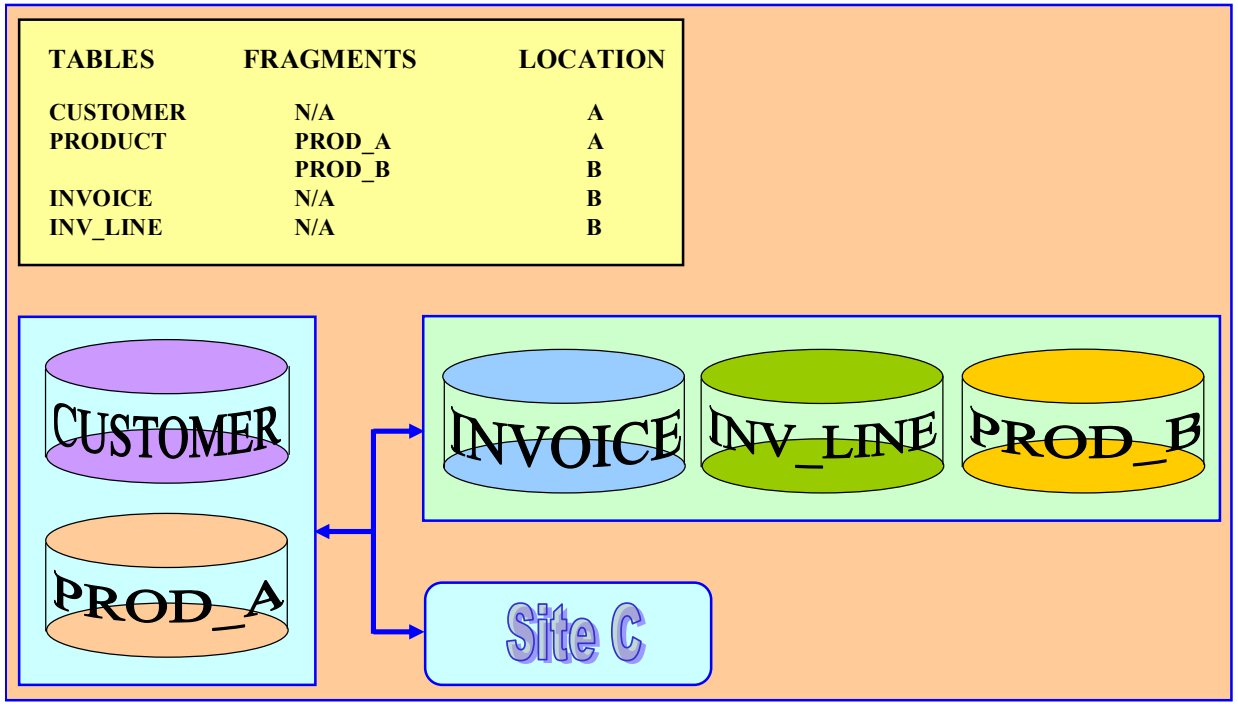


```
SELECT      *
   FROM     INVOICE
   WHERE    INV_TOTAL < 1000;
```

This SQL sequence represents a *remote request*.

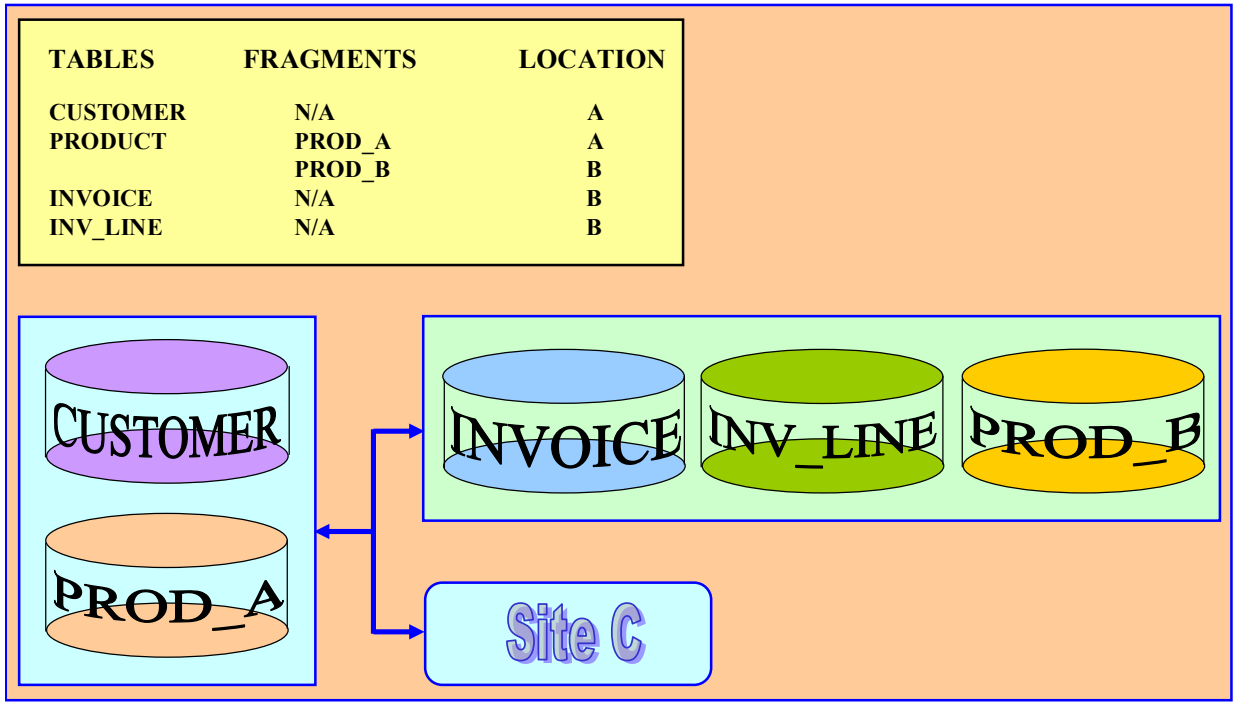


```
SELECT      *
   FROM     PRODUCT
   WHERE    PROD_QOH < 10;
```

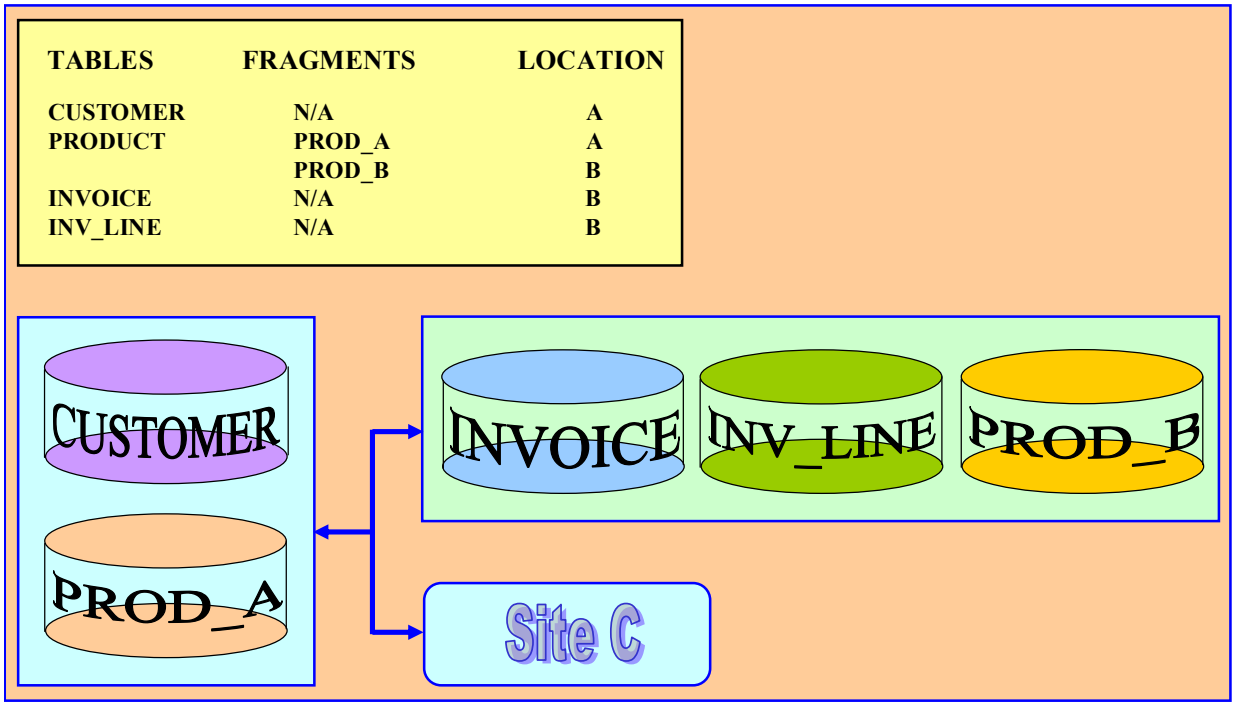


```
SELECT      *
   FROM     PRODUCT
   WHERE    PROD_QOH < 10;
```

This SQL sequence represents a *distributed request*. Note that the distributed request is required when a single request must access two DP sites. The PRODUCT table is composed of two fragments, PRO_A and PROD_B, which are located in sites A and B, respectively.



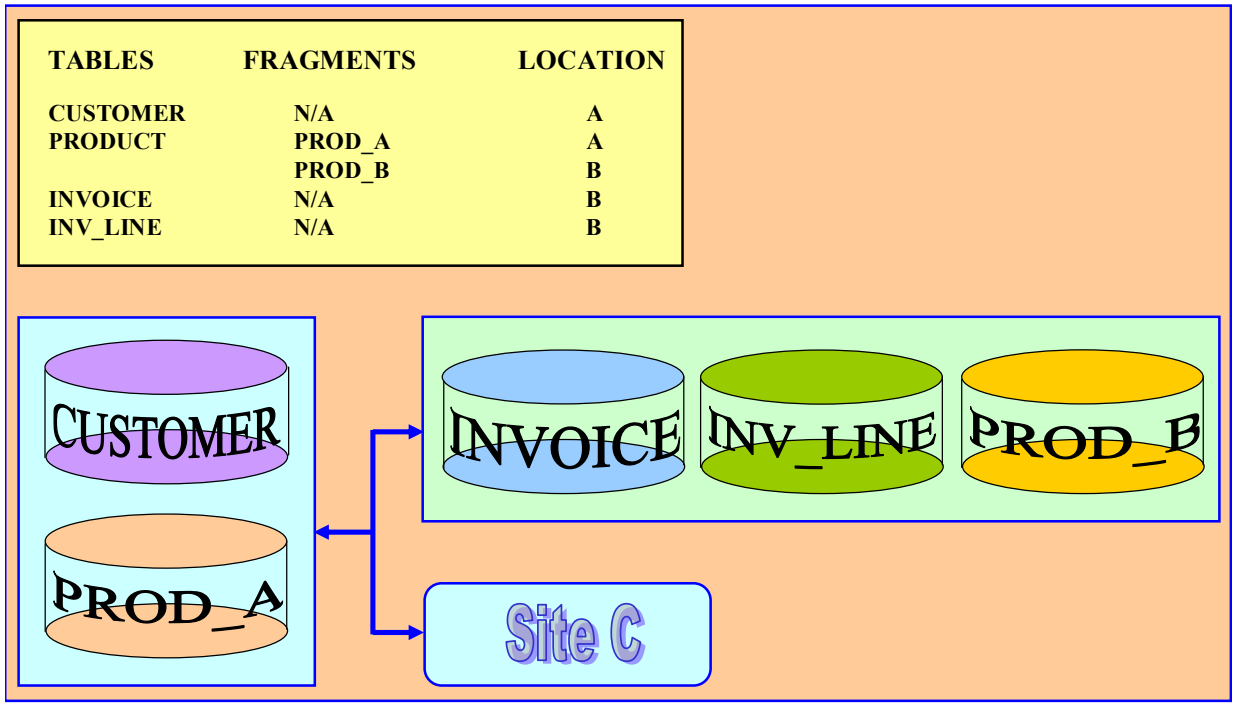
```
BEGIN WORK;
        UPDATE CUSTOMER
                SET CUS_BALANCE = CUS_BALANCE + 100
                        WHERE CUS_NUM='10936';
        INSERT INTO INVOICE(INV_NUM, CUS_NUM, INV_DATE, INV_TOTAL)
                VALUES ('986391', '10936', '15-FEB-2016', 100);
        INSERT INTO INVLINE(INV_NUM, PROD_CODE, LINE_PRICE)
                VALUES ('986391', '1023', 100);
    UPDATE PRODUCT
        SET PROD_QOH = PROD_QOH - 1
                WHERE PROD_CODE = '1023';
        COMMIT WORK;
```



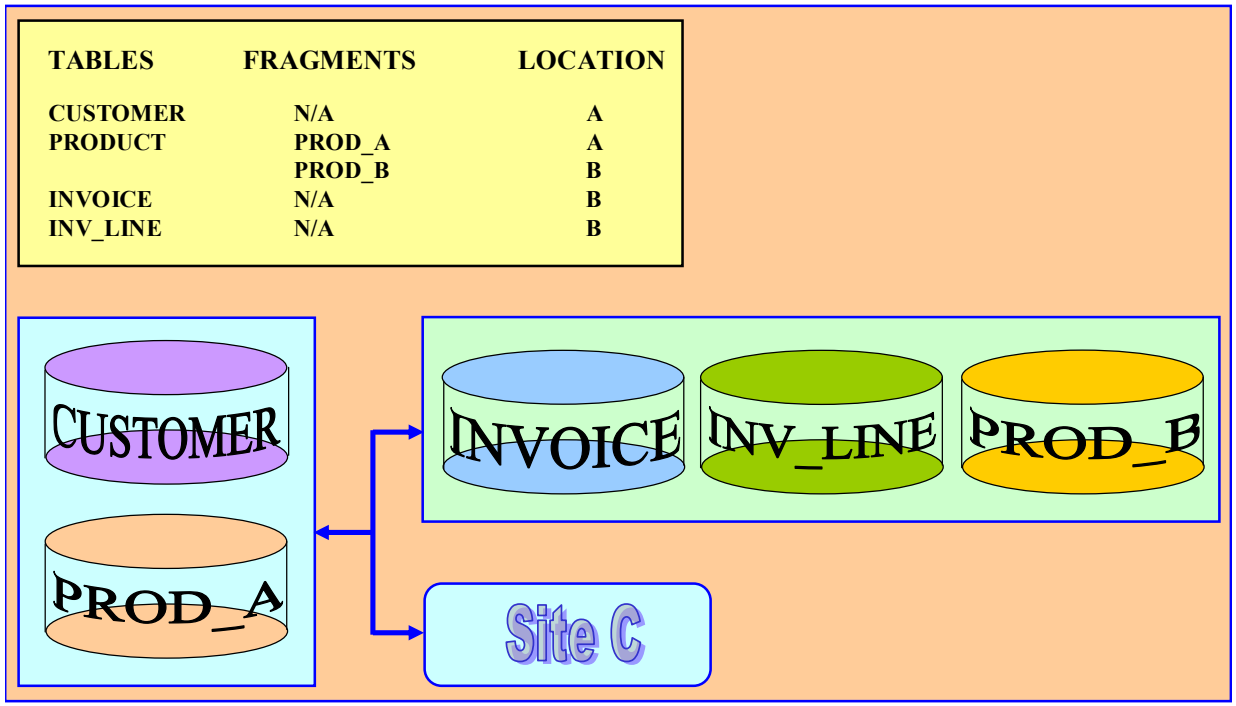
```
BEGIN WORK;
        UPDATE CUSTOMER
              SET CUS_BALANCE = CUS_BALANCE + 100
                    WHERE CUS_NUM='10936';
        INSERT INTO INVOICE(INV_NUM, CUS_NUM, INV_DATE, INV_TOTAL)
              VALUES ('986391', '10936', '15-FEB-2016', 100);
        INSERT INTO INVLINE(INV_NUM, PROD_CODE, LINE_PRICE)
              VALUES ('986391', '1023', 100);
    UPDATE PRODUCT
        SET PROD_QOH = PROD_QOH - 1
              WHERE PROD_CODE = '1023';
        COMMIT WORK;
```

This SQL sequence represents a *distributed request*.

Note that UPDATE CUSTOMER and the two INSERT statements only require remote request capabilities. However, the entire transaction must access more than one remote DP site, so we also need distributed transaction capability. The last UPDATE PRODUCT statement accesses two remote sites because the PRODUCT table is divided into two fragments located at two remote DP sites. Therefore, the transaction as a whole requires distributed request capability.

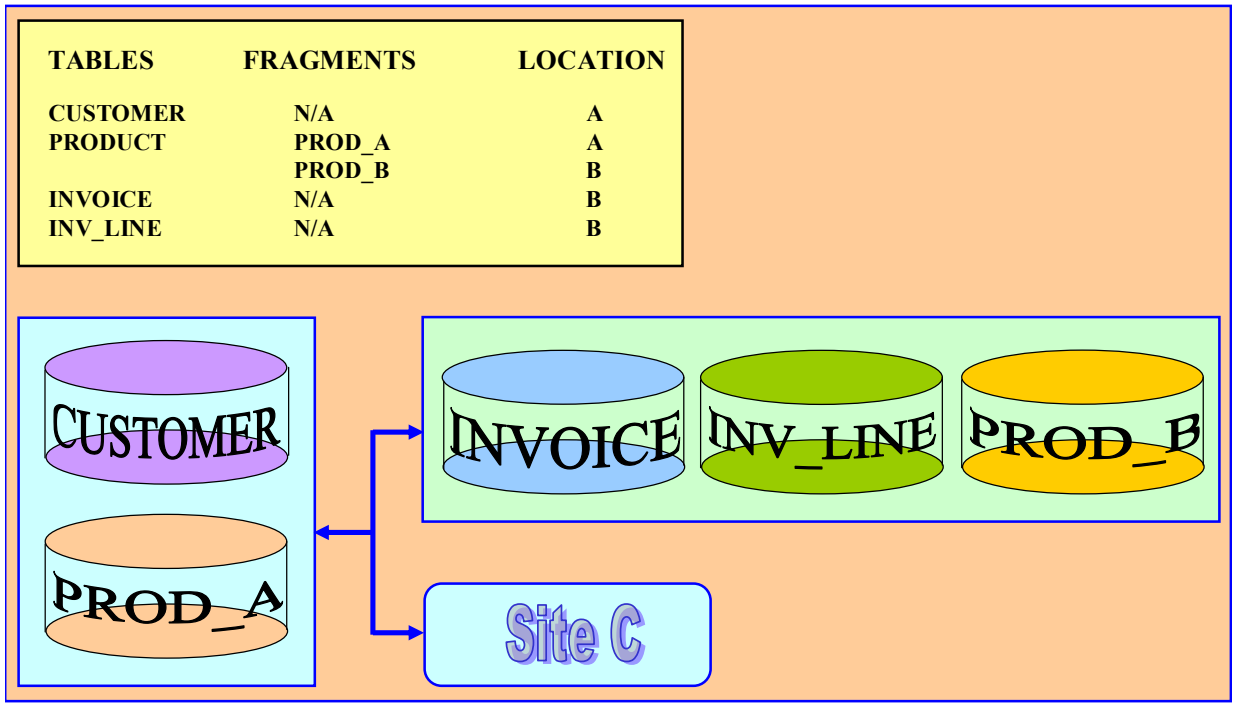


```
BEGIN WORK;
    INSERT CUSTOMER(CUS_NUM, CUS_NAME, CUS_ADDRESS, CUS_BAL)
        VALUES ('34210','Victor Ephanor', '123 Main St', 0.00);
    INSERT INTO INVOICE(INV_NUM, CUS_NUM, INV_DATE, INV_TOTAL)
        VALUES ('986434', '34210', '10-AUG-2016', 2.00);
    COMMIT WORK;
```



```
BEGIN WORK;
    INSERT CUSTOMER(CUS_NUM, CUS_NAME, CUS_ADDRESS, CUS_BAL)
        VALUES ('34210','Victor Ephanor', '123 Main St', 0.00);
    INSERT INTO INVOICE(INV_NUM, CUS_NUM, INV_DATE, INV_TOTAL)
        VALUES ('986434', '34210', '10-AUG-2016', 2.00);
    COMMIT WORK;
```

This SQL sequence represents a *distributed transaction*. Note that, in this transaction, each individual request requires only remote request capabilities. However, the transaction as a whole accesses two remote sites. Therefore, distributed request capability is required.



At A:

f.
```
SELECT    CUS_NUM, CUS_NAME, INV_TOTAL
FROM      CUSTOMER, INVOICE
WHERE     CUSTOMER.CUS_NUM = INVOICE.CUS_NUM;
```

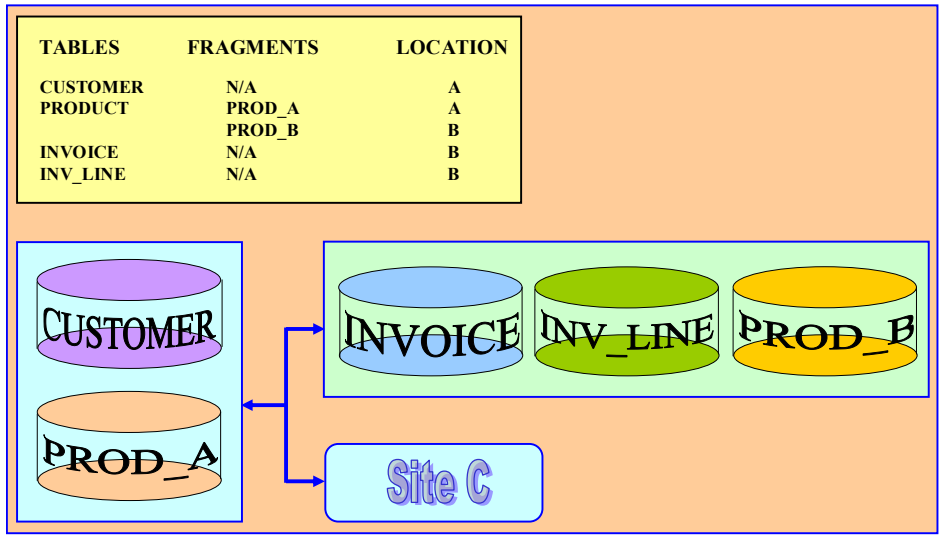
This SQL sequence represents a *distributed request*. Note that the request accesses two DP sites, one local and one remote. Therefore distributed capability is needed.

g.
```
SELECT    *
FROM      INVOICE
WHERE     INV_TOTAL > 1000;
```

This SQL sequence represents a *remote request*, because it accesses only one remote DP site.

h.
```
SELECT    *
FROM      PRODUCT
WHERE     PROD_QOH < 10;
```

This SQL sequence represents a *distributed request*. In this case, the PRODUCT table is partitioned between two DP sites, A and B. Although the request accesses only one remote DP site, it accesses a table that is partitioned into two fragments: PROD-A and PROD-B. **A single request can access a partitioned table only if the DBMS supports distributed requests.**



At B:

i.
```
SELECT    *
FROM      CUSTOMER;
```

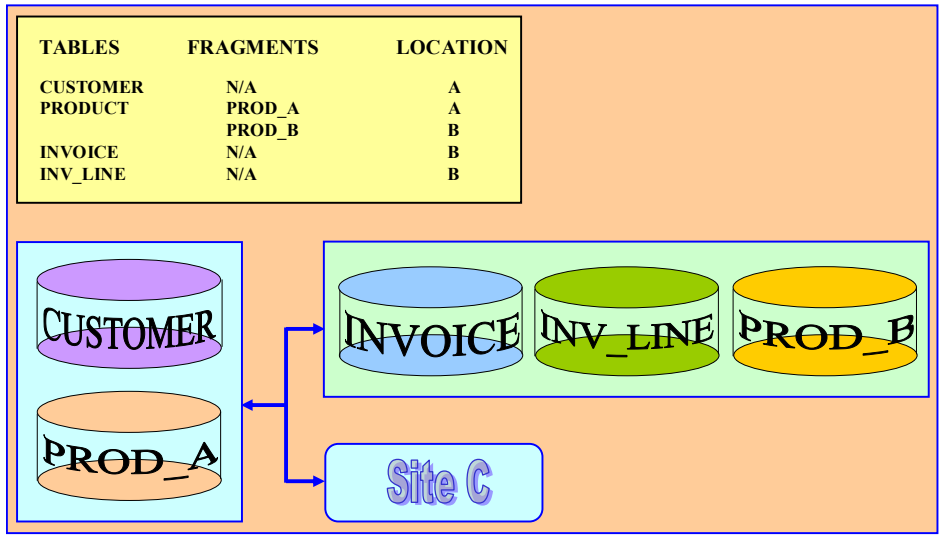
This SQL sequence represents a *remote request*.

```
SELECT    CUS_NAME, INV_TOTAL
  FROM    CUSTOMER, INVOICE
  WHERE   INV_TOTAL > 1000 AND CUSTOMER.CUS_NUM = INVOICE.CUS_NUM;
```

This SQL sequence represents a *distributed request*.

k.
```
SELECT    *
FROM      PRODUCT
WHERE     PROD_QOH < 10;
```

This SQL sequence represents a *distributed request*. (See explanation for part h.)



A fully distributed database management system must perform all of the functions of a centralized database management system (DBMS). What are these functions?

A fully distributed database management system must perform all of the functions of a centralized database management system (DBMS). What are these functions?

ANSWER: 1. Receive the request of an application or end user.

2. Validate, analyze, and decompose the request. The request might include mathematical and logical operations such as the following: Select all customers with a balance greater than $1,000. The request might require data from only a single table, or it might require access to several tables.
3. Map the request's logical-to-physical data components.
4. Decompose the request into several disk I/O operations.
5. Search for, locate, read, and validate the data.
6. Ensure database consistency, security, and integrity.
7. Validate the data for the conditions, if any, specified by the request.
8. Present the selected data in the required format.

70. Explain the difference between homogeneous and heterogeneous distributed database management systems (DDBMS).

70. Explain the difference between homogeneous and heterogeneous distributed database management systems (DDBMS).

ANSWER: Homogeneous DDBMSs integrate multiple instances of the same DBMS over a network—for example, multiple instances of Oracle 11g running on different platforms. In contrast, heterogeneous DDBMSs integrate different types of DBMSs over a network, but all support the same data model. A fully heterogeneous DDBMS will support different DBMSs, each one supporting a different data model, running under different computer systems.

Describe performance transparency and heterogeneity transparency.

Describe performance transparency and heterogeneity transparency.

ANSWER: Performance transparency allows the system to perform as if it were a centralized DBMS. The system will not suffer any performance degradation due to its use on a network or because of the network's platform differences. Performance transparency also ensures that the system will find the most cost-effective path to access remote data. The system should be able to "scale out" in a transparent manner, or increase performance capacity by adding more transaction or data-processing nodes, without affecting the overall performance of the system.

Explain the three types of operations defined by the DO-UNDO-REDO protocol.

Explain the three types of operations defined by the DO-UNDO-REDO protocol.

ANSWER:

1. DO performs the operation and records the "before" and "after" values in the transaction log.
2. UNDO reverses an operation, using the log entries written by the DO portion of the sequence.
3. REDO redoes an operation, using the log entries written by the DO portion of the sequence.

The following data structure and constraints exist for a magazine publishing company.

a. **The company publishes one regional magazine each in Florida (FL), South Carolina (SC), Georgia (GA), and Tennessee (TN).**

b. **The company has 300,000 customers (subscribers) distributed throughout the four states listed in Part 2a.**

c. **On the first of each month, an annual subscription INVOICE is printed and sent to each customer whose subscription is due for renewal. The INVOICE entity contains a REGION attribute to indicate the customer's state of residence (FL, SC, GA, TN):**

CUSTOMER (CUS_NUM, CUS_NAME, CUS_ADDRESS, CUS_CITY, CUS_STATE, CUS_ZIP, CUS_SUBSDATE)

INVOICE (INV_NUM, INV_REGION, CUS_NUM, INV_DATE, INV_TOTAL)

The company is aware of the problems associated with centralized management and has decided that it is time to decentralize the management of the subscriptions in its four regional subsidiaries. Each subscription site will handle its own customer and invoice data. The management at company headquauters, however, will have access to customer and invoice data to generate annual reports and to issue ad hoc queries, such as:

- **List all current customers by region.**
- **List all new customers by region.**
- **Report all invoices by customer and by region.**

Given these requirements, how must you partition the database?

The CUSTOMER table must be partitioned horizontally by state. (We show the partitions in the answer to 3c.)

Given the scenario and the requirements in Problem 2, answer the following questions:

a. What recommendations will you make regarding the type and characteristics of the required database system?

b. What type of data fragmentation is needed for each table?

Given the scenario and the requirements in Problem 2, answer the following questions:

a. What recommendations will you make regarding the type and characteristics of the required database system?

The Magazine Publishing Company requires a distributed system with distributed database capabilities. The distributed system will be distributed among the company locations in South Carolina, Georgia, Florida, and Tennessee.

The DDBMS must be able to support distributed transparency features, such as fragmentation transparency, replica transparency, transaction transparency, and performance transparency. Heterogeneous capability is not a mandatory feature since we assume there is no existing DBMS in place and that the company wants to standardize on a single DBMS.

b. What type of data fragmentation is needed for each table?

The database must be horizontally partitioned, using the STATE attribute for the CUSTOMER table and the REGION attribute for the INVOICE table.

What must be the criteria used to partition each database?

What must be the criteria used to partition each database?

The following fragmentation segments reflect the criteria used to partition each database:

Horizontal Fragmentation of the CUSTOMER Table by State

Fragment Name	Location	Condition	Node name
C1	Tennessee	CUS_STATE = 'TN'	NAS
C2	Georgia	CUS_STATE = 'GA'	ATL
C3	Florida	CUS_STATE = 'FL'	TAM
C4	South Carolina	CUS_STATE = 'SC'	CHA

Horizontal Fragmentation of the INVOICE Table by Region

Fragment Name	Location	Condition	Node name
I1	Tennessee	REGION_CODE = 'TN'	NAS
I2	Georgia	REGION_CODE = 'GA'	ATL
I3	Florida	REGION_CODE = 'FL'	TAM
I4	South Carolina	REGION_CODE = 'SC'	CHA

Design the database fragments. Show an example with node names, location, fragment names, attribute names, and demonstration data.

Design the database fragments. Show an example with node names, location, fragment names, attribute names, and demonstration data.

Note the following fragments:

Fragment C1 **Location: Tennessee** **Node: NAS**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_CITY	CUS_STATE	CUS_SUB_DATE
10884	James D. Burger	123 Court Avenue	Memphis	TN	8-DEC-16
10993	Lisa B. Barnette	910 Eagle Street	Nashville	TN	12-MAR-17

Fragment C2 **Location: Georgia** **Node: ATL**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_CITY	CUS_STATE	CUS_SUB_DATE
11887	Ginny E. Stratton	335 Main Street	Atlanta	GA	11-AUG-16
13558	Anna H. Ariona	657 Mason Ave.	Dalton	GA	23-JUN-17

Fragment C3 **Location: Florida** **Node: TAM**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_CITY	CUS_STATE	CUS_SUB_DATE
10014	John T. Chi	456 Brent Avenue	Miami	FL	18-NOV-16
15998	Lisa B. Barnette	234 Ramala Street	Tampa	FL	23-MAR-17

Fragment C4 **Location: South Carolina** **Node: CHA**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_CITY	CUS_STATE	CUS_SUB_DATE
21562	Thomas F. Matto	45 N. Pratt Circle	Charleston	SC	2-DEC-16
18776	Mary B. Smith	526 Boone Pike	Charleston	SC	28-OCT-17

Fragment I1 **Location: Tennessee** **Node: NAS**

INV_NUM	REGION_CODE	CUS_NUM	INV_DATE	INV_TOTAL
213342	TN	10884	1-NOV-15	45.95
209987	TN	10993	15-FEB-16	45.95

Fragment I2 **Location: Georgia** **Node: ATL**

INV_NUM	REGION_CODE	CUS_NUM	INV_DATE	INV_TOTAL
198893	GA	11887	15-AUG-15	70.45
224345	GA	13558	1-JUN-16	45.95

Fragment I3 **Location: Florida** **Node: TAM**

INV_NUM	REGION_CODE	CUS_NUM	INV_DATE	INV_TOTAL
200915	FL	10014	1-NOV-15	45.95
231148	FL	15998	1-MAR-16	24.95

Fragment I4 **Location: South Carolina** **Node: CHA**

INV_NUM	REGION_CODE	CUS_NUM	INV_DATE	INV_TOTAL
243312	SC	21562	15-NOV-15	45.95
231156	SC	18776	1-OCT-16	45.95

What type of distributed database operations must be supported at each remote site?

What type of distributed database operations must be supported at each remote site?

To answer this question, you must first draw a map of the locations, the fragments at each location, and the type of transaction or request support required to access the data in the distributed database.

	Node				
Fragment	**NAS**	**ATL**	**TAM**	**CHA**	**Headquarters**
CUSTOMER	C1	C2	C3	C4	
INVOICE	I1	I2	I3	I4	
Distributed Operations Required	none	none	none	none	distributed request

Given the problem's specifications, you conclude that no interstate access of CUSTOMER or INVOICE data is required. Therefore, no distributed database access is required in the four nodes. For the headquarters, the manager wants to be able to access the data in all four nodes through a single SQL request. Therefore, the DDBMS must support distributed requests.