05- DDBMS Design

School of Computer Science
University of Windsor

Agenda

- Lecture
  - Distributed Database Design
  - Data Allocation
  - Fragmentation
- Lab 2

Announcements - Submission deadlines

- **Lab 2**– Sec 2: Feb 8; Sec 3: Feb 9; Sec 4: Feb 10, 11:59 PM

  Note: Not for undergrad students

- **Assignment 2- Hadoop LinkedIn Learning Course**
- Certificate (2%) Due : Sec 2: Feb 13; Sec 3: Feb 14; Sec 4: Feb 15
- Quiz (3%) Next day of submission

Introductory Questions

What important things need to be considered in data allocation?

What is fragmentation?

Why is fragmentation important in DDBMS?

How can we do fragmentation?

Distributed Database Design

Fragmentation

A relation may be divided into a number of sub-relations, which are then distributed. .

Two main types:

- Horizontal fragments are subsets of tuples.
- Vertical fragments are subsets of attributes.

Uses qualitative information.

Allocation

Each fragment is stored at the site with "optimal" distribution.

Uses quantitative information.

Replication

The DDBMS may maintain a copy of a fragment at several different sites.

80/20 Rule in D-DBMS

“**20 percent** of the active user transactions are responsible for **80 percent** of the data access” [1]

Distributed Database Design

The definition and allocation of fragments must be based on how the database is to be used. This involves analyzing both **quantitative** and **qualitative** information from transactions.

- ✓ The **qualitative information** is used for ***fragmentation***, and may include information about the transactions that are executed, such as:
  - the relations, attributes, and tuples accessed;
  - the type of access (read or write);
  - the predicates of read operations.

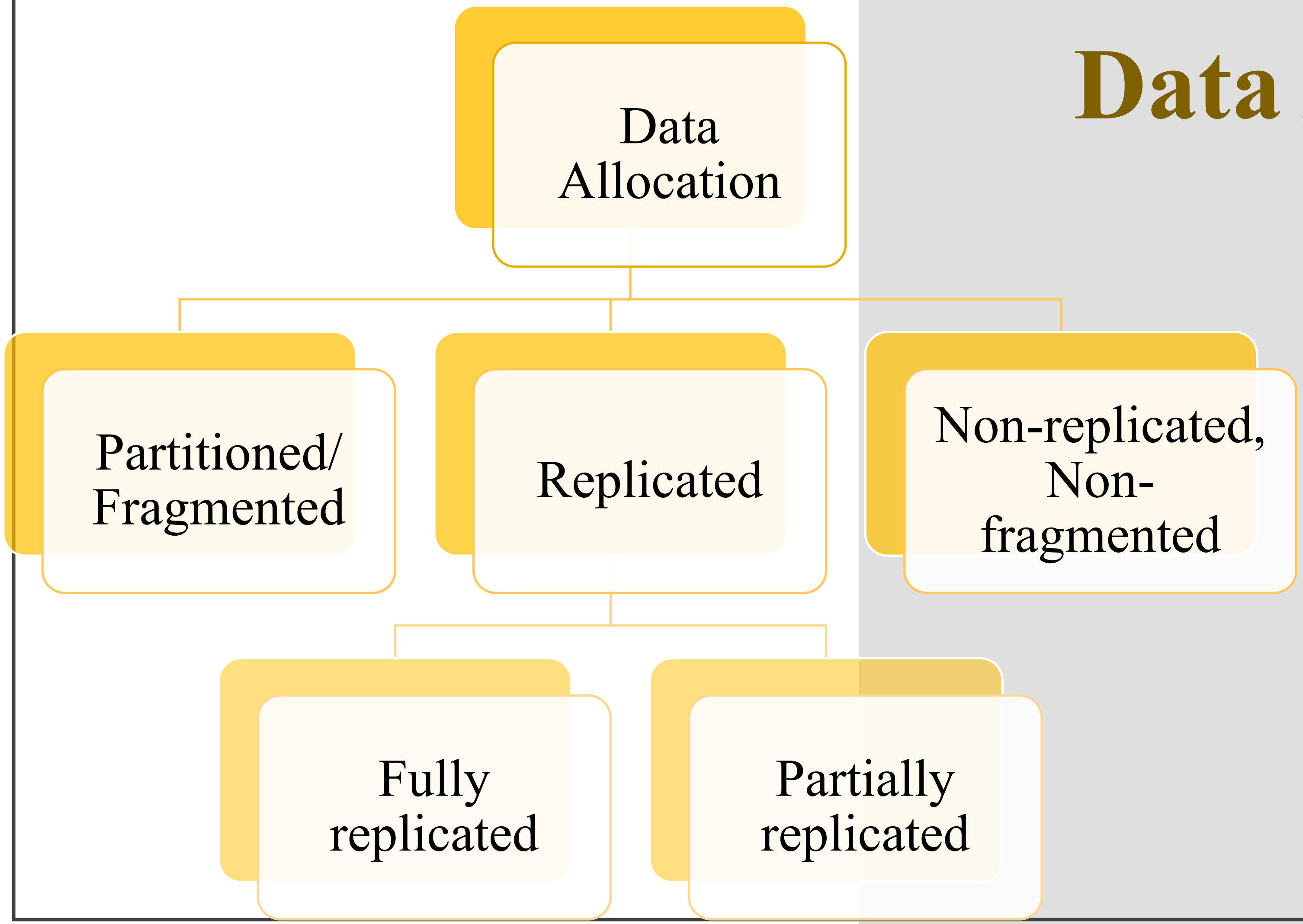
- ✓ The **quantitative information** is used in ***allocation***, and may include:
  - the frequency with which a transaction is run;
  - the site from which a transaction is run;
  - the performance criteria for transactions.

Fragments Allocation Objectives

Definition and allocation of fragments are carried out strategically to achieve following objectives:

1. **Locality of reference**: Where possible, data should be stored close to where it is used. If a fragment is used at several sites, it may be advantageous to store copies of the fragment at these sites.
2. **Improved reliability and availability**: Reliability and availability are improved by replication.
3. **Acceptable performance**. Bad allocation may result in bottlenecks and underutilization of resources.
4. **Balanced storage capacities and costs:** Consideration should be given to the availability and cost of storage at each site, so that cheap mass storage can be used where possible.
5. **Minimal communication costs:** Consideration should be given to the cost of remote requests.

Data Allocation



Comparison of Strategies for Data Allocation

	LOCALITY OF REFERENCE	RELIABILITY AND AVAILABILITY	PERFORMANCE	STORAGE COSTS	COMMUNICATION COSTS
Centralized	**Low**	**Low**	**Unsatisfactory**	**Low**	**High**

Comparison of Strategies for Data Allocation

	LOCALITY OF REFERENCE	RELIABILITY AND AVAILABILITY	PERFORMANCE	STORAGE COSTS	COMMUNICATION COSTS
Centralized	Low	Low	Unsatisfactory	Low	High
Fragmented	**High**	**Low**	**Satisfactory**	**Low**	**Low**

Comparison of Strategies for Data Allocation

	LOCALITY OF REFERENCE	RELIABILITY AND AVAILABILITY	PERFORMANCE	STORAGE COSTS	COMMUNICATION COSTS
Centralized	Lowest	Lowest	Unsatisfactory	Lowest	Highest
Fragmented	High	Low	Satisfactory	Lowest	Low
Complete replication	**High**	**High**	**Best for read**	**High**	**High for update; low for read**

Comparison of Strategies for Data Allocation

	LOCALITY OF REFERENCE	RELIABILITY AND AVAILABILITY	PERFORMANCE	STORAGE COSTS	COMMUNICATION COSTS
Centralized	Low	Low	Unsatisfactory	Low	High
Fragmented	High	Low	Satisfactory	Low	Low
Complete replication	High	High	Best for read	High	High for update; low for read
Selective replication	**High**	**Low**	**Satisfactory**	**Average**	**Low**

Test your understanding

A distributed database can use which of the following strategies?

A. Totally centralized at one location and accessed by many sites

B. Partially or completely replicated across sites

C. Partitioned into fragments at different sites

D. All of the above

Why Fragmentation?

Usage: Applications work with views rather than entire relations (tables).

Parallelism: transaction can be divided into several subqueries that operate on fragments.

Efficiency: Data is stored close to where it is most frequently used.

Data that is not needed by local applications is not stored.

Security: Data not required by local applications is not stored and so not available to unauthorized users.

Disadvantages of Fragmentation

Performance: The performance of global applications that require data from several fragments located at different sites may be slower.

Integrity: Integrity control may be more difficult if data and functional dependencies are fragmented and located at different sites.

Correctness of Fragmentation

Fragmentation cannot be carried out randomly. There are **three rules** that must be followed during fragmentation. This rule is necessary to ensure that there is no loss of data during fragmentation.

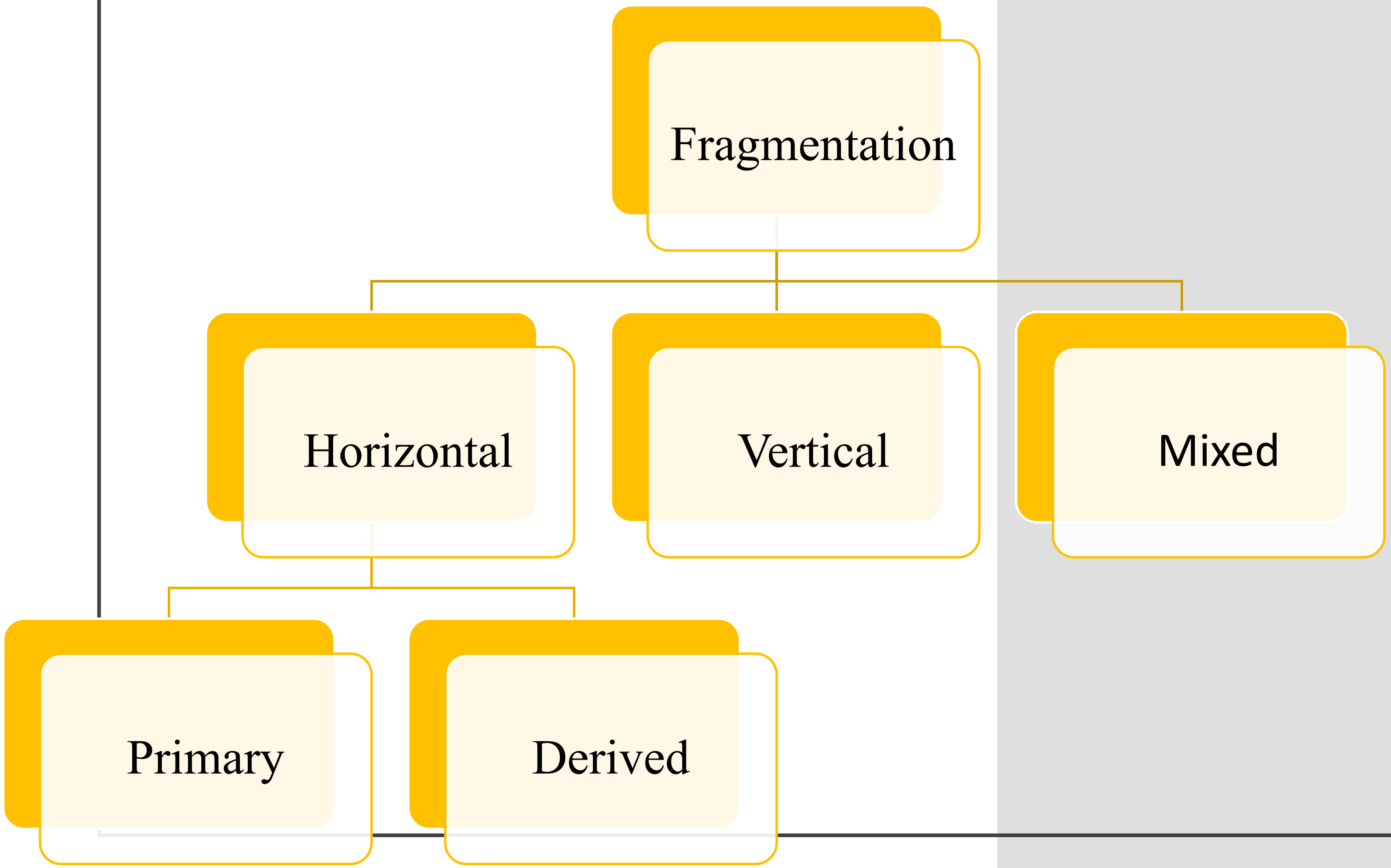
Completeness: If relation R is decomposed into fragments R_1, R_2, ... R_n, each data item that can be found in R must appear in at least one fragment.

Reconstruction: Must be possible to define a relational operation that will reconstruct R from the fragments.
Reconstruction for horizontal fragmentation is **Union** operation and **Join** for vertical .

Disjointness: If data item d_i appears in fragment R_i, then it should not appear in any other fragment.

- Exception: vertical fragmentation, where primary key attributes must be repeated to allow reconstruction.

Types of Fragmentation



Instance of the DreamHome Rental Database.

Branch

BranchNo	Street	City	Postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

PropertyForRent

PropertyNo	Street	City	Postcode	Type	Rooms	Rent	OwnerNo	StaffNo	BranchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

Staff

StaffNo	fName	lName	Position	Sex	DOB	Salary	BranchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Client

clientNo	fName	lName	telNo	prefType	maxRent	eMail
CR76	John	Kay	0207-774-5632	Flat	425	ohn.kay@gmail.com
CR56	Aline	Stewart	0141-848-1825	Flat	350	astewart@hotmail.com
CR74	Mike	Ritchie	01475-392178	House	750	mritchie01@yahoo.co.uk
CR62	Mary	Tregear	01224-196720	Flat	600	maryt@hotmail.co.uk

Instance of the DreamHome Rental Database.

PrivateOwner

ownerNo	fName	IName	address	telNo	eMail	password
CO46	Joe	Keogh	2 Fergus Dr, Aberdeen AB2 7SX	01224-861212	jkeogh@lhh.com	********
CO87	Carol	Farrel	6 Achray St, Glasgow G32 9DX	0141-357-7419	cfarrel@gmail.com	********
CO40	Tina	Murphy	63 Well St, Glasgow G42	0141-943-1728	tinam@hotmail.com	********
CO93	Tony	Shaw	12 Park Pl, Glasgow G4 OQR	0141-225-7025	tony.shaw@ark.com	********

Registration

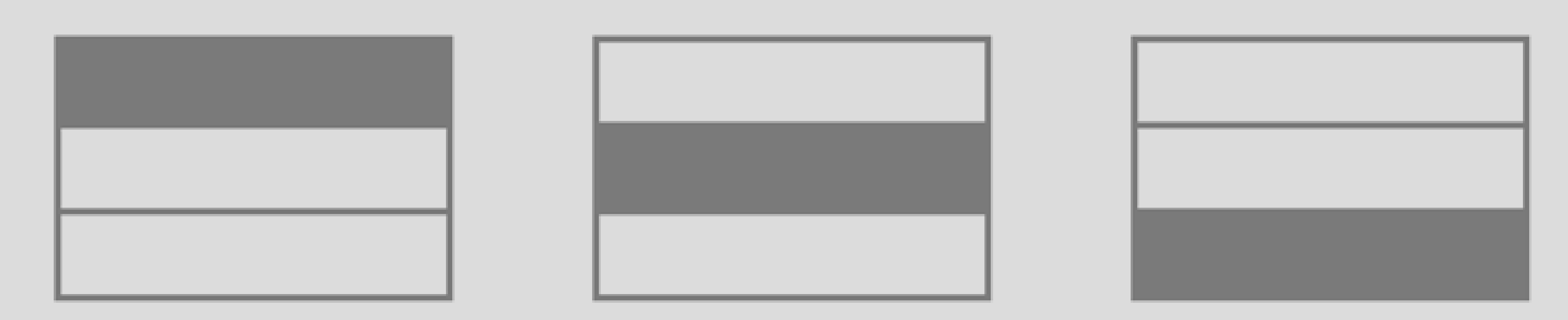
clientNo	branchNo	staffNo	dateJoined
CR76	BOOS	SL41	2-Jan- 13
CR56	8003	SG37	1 1-Apr-12
CR74	8003	SG37	16-Nov-ll
CR62	B007	SA9	7-03-12

Viewing

clientNo	propertyNo	ViewDate	Comment
CR56	PA14	24-May-13	too small
CR76	PG4	20-Apr-13	too remote
CR56	PG4	26-May-13	
CR62	PA14	14-May-13	no dining room
CR56	PG36	28-Apr-13	

Horizontal Fragmentation

Consists of a subset of the tuples of a relation.



Primary Horizontal Fragmentation (PHF)

A horizontal fragment is produced by specifying a predicate that performs a restriction on the tuples in a **single** relation.

Defined using ***Selection*** operation of relational algebra:

Given a relation R, a horizontal fragment is defined as:

$$\sigma_p(R)$$

where p is a predicate based on one or more attributes of the relation.

PropertyForRent

PropertyNo	Street	City	Postcode	Type	Rooms	Rent	OwnerNo	StaffNo	BranchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	C046	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	C087	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	C040		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	C093	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	C087	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	C093	SG14	B003

The predicates may be simple or complex:

1. **Simple predicates**: Given a table/relation R with set of attributes $[A_1, A_2, A_3, A_4, \ldots, A_n]$, a simple predicate Pi can be expressed as follows;

   $$P_i : A_j \ \theta \ \text{Value}$$

   Where θ can be any of the symbols in the set $\{\leq, \geq, \neq, <, >, =\}$, a value can be any value stored in the table for the attributed A_i.

   Example : attribute, comparison-operator value; **Type='House'**

2. Complex (Set of simple) predicates: A relation fragmented using Boolean combination of simple predicates;

Example : P={Type='House' AND Rooms>2}

Horizontal Fragmentation Strategy

Predicate Selectivity Criteria:

1. **Access frequency**: An examination of the predicates (or search conditions) used by transactions or queries in the applications.
2. **Complete**: A set of predicates is complete if and only if any two tuples in the same fragment are referenced with the same probability by any transaction.

   Example: If the only requirement is to select tuples from PropertyForRent based on the property type,

   set {type = 'House', type = 'Flat'} is complete

   whereas the set {type = 'House'} is not complete
3. **Relevant (minimal)**: A predicate is relevant if there is at least one transaction that accesses the resulting fragments differently.

   Example: city = 'Aberdeen' is not relevant.

Primary Horizontal Fragmentation (PHF)

When we fragment any relation horizontally, we use single condition or set of simple predicates to filter the data.
Given a relation R and set of simple predicates, we can fragment a relation horizontally as follows;

$$R_i = \sigma_{pi}(R),\ 1 \le i \le n$$

where Pi is the set of simple predicates, also called as a Min-term predicate

For example:

$$P_1 = \sigma_{\text{type='House'}}(\text{PropertyForRent})$$

$$P_2 = \sigma_{\text{type='Flat'}}(\text{PropertyForRent})$$

PropertyForRent

PropertyNo	Street	City	Postcode	Type	Rooms	Rent	OwnerNo	StaffNo	BranchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	C046	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	C087	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	C040		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	C093	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	C087	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	C093	SG14	B003

Fragment P_1

PropertyNo	Street	City	Postcode	Type	Rooms	Rent	OwnerNo	StaffNo	BranchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	C046	SA9	B007
PG21	18 Dale Rd	Glasgow	G12	House	5	600	C087	SG37	B003

Fragment P_2

PropertyNo	Street	City	Postcode	Type	Rooms	Rent	OwnerNo	StaffNo	BranchNo
PL94	6 Argyll St	London	NW2	Flat	4	400	C087	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	C040		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	C093	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	C093	SG14	B003

Correctness of Fragmentation

Fragment P_2

PropertyNo	Street	City	Postcode	Type	Rooms	Rent	OwnerNo	StaffNo	BranchNo
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

Fragment P_1

Property No	Street	City	Postcode	Type	Rooms	Rent	Owner No	Staff No	Branch No
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003

Completeness: If relation R is decomposed into fragments R_1, R_2, ... R_n, each data item that can be found in R must appear in at least one fragment.

Each tuple in the relation appears in either fragment P_1 or P_2.

Reconstruction: Must be possible to define a relational operation that will reconstruct R from the fragments.

The PropertyForRent relation can be reconstructed from the fragments using the Union operation:

$$P_1 \cup P_2 = \text{PropertyForRent}$$

Disjointness: If data item d_i appears in fragment R_i, then it should not appear in any other fragment.

The fragments are disjoint; there can be no property type that is both 'House' and 'Flat'.

Min-Term Predicates

Min-term predicate: A conjunction of simple and negated simple predicates

Given R and $P_r=\{p_{1,} p_2 \ldots,p_m\}$

define $M=\{m_1,m_2,\ldots,m_r\}$ as

$M=\{ m_i|m_i = \wedge_{pj\in Pr}\, p_j^* \}$, $1\leq j\leq m$, $1\leq i\leq z$ where $p_j^* = p_j$ or $p_j^* = \neg(p_j)$.

Example

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng.
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

ASG

ENO	PNO	RESP	DUR
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E8	P3	Manager	40

PROJ

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York
P3	CAD/CAM	250000	New York
P4	Maintenance	310000	Paris

PAY

TITLE	SAL
Elect. Eng.	40000
Syst. Anal.	34000
Mech. Eng.	27000
Programmer	24000

The following are some of the possible simple predicates that can be defined on PAY.

$p1$: TITLE = "Elect. Eng."
$p2$: TITLE = "Syst. Anal."
$p3$: TITLE = "Mech. Eng."
$p4$: TITLE = "Programmer"
$p5$: SAL $\leq$ 30000

The following are some of the minterm predicates that can be defined based on these simple predicates.

m1 : TITLE = "Elect. Eng." $\wedge$ SAL $\leq$ 30000
m2 : TITLE = "Elect. Eng." $\wedge$ SAL $>$ 30000
m3 : $\neg$(TITLE = "Elect. Eng.") $\wedge$ SAL $\leq$ 30000
m4 : $\neg$(TITLE = "Elect. Eng.") $\wedge$ SAL $>$ 30000
m5 : TITLE = "Programmer" $\wedge$ SAL $\leq$ 30000
m6 : TITLE = "Programmer" $\wedge$ SAL $>$ 30000

Derived Horizontal Fragmentation (DHF)

Some applications may involve a join of two or more relations. A horizontal fragment that is based on the horizontal fragmentation of a parent relation.

Derived fragmentation is defined using the **Semijoin operation (⊲)** of the relational algebra.

Given a child relation R and parent S, the derived fragmentation of R is defined as:

$$R_i = R \lhd_f S_i, \qquad 1 \leq i \leq w$$

where w is the number of horizontal fragments defined on S and f is the join attribute.

For example:

$S_3 = \sigma_{branchNo='B003'}(Staff)$

$S_4 = \sigma_{branchNo='B005'}(Staff)$

$S_5 = \sigma_{branchNo='B007'}(Staff)$

Staff

StaffNo	fName	lName	Position	Sex	DOB	Salary	BranchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Could use derived fragmentation for Property:

$P_i = \text{PropertyForRent} \lhd_{staffNo} S_i,$

$3 \leq i \leq 5$

Derived fragmentation of PropertyForRent based on Staff

Fragment P_3

PropertyNo	Street	City	Postcode	Type	Rooms	Rent	OwnerNo	StaffNo	BranchNo
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40	SG14	B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

Fragment P_4

PropertyNo	Street	City	Postcode	Type	Rooms	Rent	OwnerNo	StaffNo	BranchNo
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005

Fragment P_5

PropertyNo	Street	City	Postcode	Type	Rooms	Rent	OwnerNo	StaffNo	BranchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007

Test your understanding

A(n) ______ database stores each database fragment at a single site.

A. partially replicated

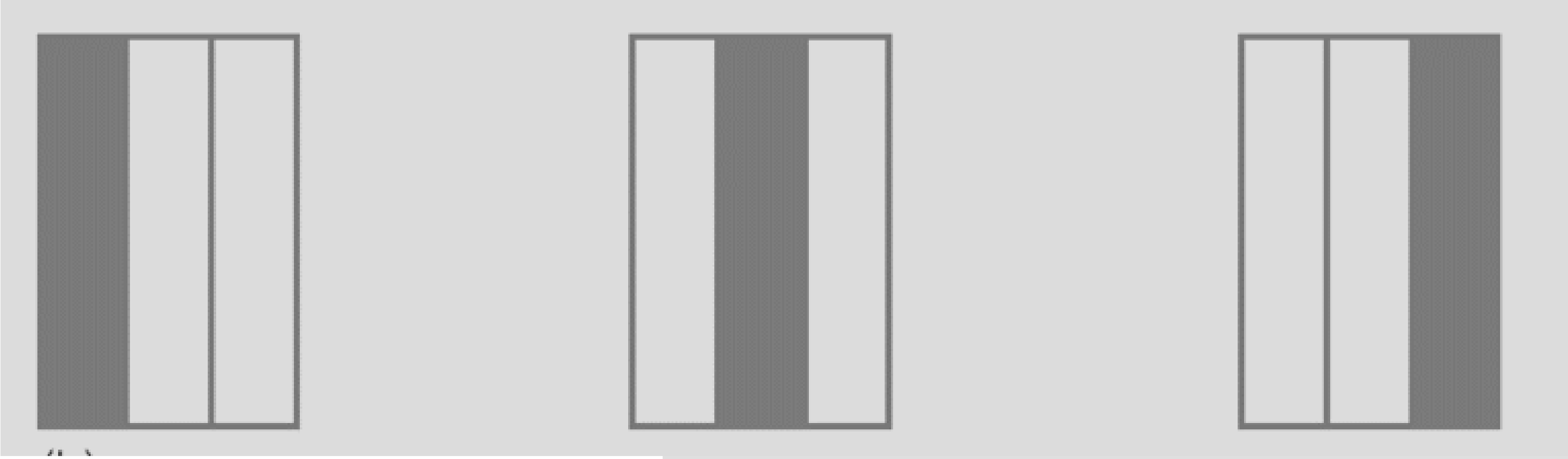
B. unreplicated

C. fully replicated

D. partitioned

Vertical Fragmentation

Vertical fragmentation groups together the attributes in a relation that are used jointly by the important transactions.



Vertical Fragmentation

Consists of a subset of attributes of a relation.
Defined using ***Projection*** operation of relational algebra:
Given a relation R, a vertical fragment is defined as:

$$\Pi_{a1, \ldots, an}(R)$$

where $a_1, \ldots, a_n$ are attributes of the relation R.

Staff

StaffNo	fName	lName	Position	Sex	DOB	Salary	BranchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Example?

- The DreamHome payroll application requires the staff number, position, sex, DOB, and salary attributes of each member of staff;
- the HR department requires the staffNo, fName, lName, and branchNo attributes.

Vertical Fragmentation

Consists of a subset of attributes of a relation.

Defined using ***Projection*** operation of relational algebra:

Given a relation R, a vertical fragment is defined as:

$$\Pi_{a1, \ldots, an}(R)$$

where $a_1, \ldots, a_n$ are attributes of the relation R.

For example:

$$S_1 = \Pi_{staffNo,\ position,\ sex,\ DOB,\ salary}(Staff)$$
$$S_2 = \Pi_{staffNo,\ fName,\ lName,\ branchNo}(Staff)$$

Staff

StaffNo	fName	lName	Position	Sex	DOB	Salary	BranchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Fragment S_1

StaffNo	Position	Sex	DOB	Salary
SL21	Manager	M	1-Oct-45	30000
SG37	Assistant	F	10-Nov-60	12000
SG14	Supervisor	M	24-Mar-58	18000
SA9	Assistant	F	19-Feb-70	9000
SG5	Manager	F	3-Jun-40	24000
SL41	Assistant	F	13-Jun-65	9000

Fragment S_2

StaffNo	fName	lName	BranchNo
SL21	John	White	B005
SG37	Ann	Beech	B003
SG14	David	Ford	B003
SA9	Mary	Howe	B007
SG5	Susan	Brand	B003
SL41	Julie	Lee	B005

Correctness of Fragmentation

Completeness: If relation R is decomposed into fragments R_1, R_2, ... R_n, each data item that can be found in R must appear in at least one fragment.

Each attribute in the Staff relation appears in either fragment S_1 or S_2.

Reconstruction: Must be possible to define a relational operation that will reconstruct R from the fragments.

The Staff relation can be reconstructed from the fragments using the Natural join operation:

$$S_1 \bowtie S_2 = \text{Staff}$$

Disjointness: If data item d_i appears in fragment R_i, then it should not appear in any other fragment.

The fragments are disjoint except for the primary key, which is necessary for reconstruction.

Fragment S_1

StaffNo	Position	Sex	DOB	Salary
SL21	Manager	M	1-Oct-45	30000
SG37	Assistant	F	10-Nov-60	12000
SG14	Supervisor	M	24-Mar-58	18000
SA9	Assistant	F	19-Feb-70	9000
SG5	Manager	F	3-Jun-40	24000
SL41	Assistant	F	13-Jun-65	9000

Fragment S_2

StaffNo	fName	lName	BranchNo
SL21	John	White	B005
SG37	Ann	Beech	B003
SG14	David	Ford	B003
SA9	Mary	Howe	B007
SG5	Susan	Brand	B003
SL41	Julie	Lee	B005

The Choice of Vertical Fragmentation Strategy

1. **Grouping**:
   Merging attributes to fragments. Starts by creating as many vertical fragments as possible and then incrementally reducing the number of fragments by merging the fragments together. See [Hammer79] and [Sacca85].
2. **Splitting**:
   Dividing a relation into fragments
   Vertical fragments can be determined by establishing the **affinity** of one attribute to another.

Steps involved are:

1. Find attribute **usage matrix** of the application queries.
2. Obtain **attribute affinity matrix**
3. Use a clustering algorithm to group some attributes based on the attribute affinity matrix. This algorithm creates a **clustered affinity matrix**
4. Use a **partitioning algorithm** to partition attributes such that sets of attributes are accessed solely or for the most part by distinct set of applications

Step 1- Attribute Usage Matrix Example

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng.
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

ASG

ENO	PNO	RESP	DUR
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E8	P3	Manager	40

PROJ

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York
P3	CAD/CAM	250000	New York
P4	Maintenance	310000	Paris

PAY

TITLE	SAL
Elect. Eng.	40000
Syst. Anal.	34000
Mech. Eng.	27000
Programmer	24000

Assume that the following queries are defined to run on PROJ relation.

AP1 q1: SELECT BUDGET FROM PROJ WHERE PNO=Value;
AP2 q2: SELECT PNAME, BUDGET FROM PROJ;
AP3 q3: SELECT PNAME FROM PROJ WHERE LOC=Value;
AP4 q4: SELECT SUM(BUDGET) FROM PROJ WHERE LOC=Value;

	PNO	PNAME	BUDGET	LOC
q1	1	0	1	0
q2	0	1	1	0
q3	0	1	0	1
q4	0	0	1	1

Attribute Usage Matrix

Step 2- Attribute Affinity Matrix - Example

The attribute affinity measure between two attributes Ai and Aj of a relation R(A1, A2, . . . , An) with respect to the set of queries

$$aff(A_i, A_j) = \sum_{k|use(qk\,,Ai\,)=1 \wedge use(qk\,,Aj\,)=1|} \; \sum_{\forall Sl} refl(qk)accl(qk)$$

Where refl(qk)- the # of **reference** of accesses to attributes (Ai , Aj) for each execution of application qk at site Sl

accl(qk) - application **access** frequency measure previously defined and modified to include frequencies at different sites.

For simplicity, let us assume that refl(qk) = 1 for all qk and Sl .

If the application frequencies are

Then the **affinity measure** between attributes PNO and BUDGET can be measured as

$$\boldsymbol{aff(PNO\,,BUDGET) = \sum_{k=1\ to\ 1} \sum_{S=1\ to\ 3} accl(qk) = acc1(q1)\ +\ acc2(q1)\ +\ acc3(q1)\ =\ 45}$$

Attribute Usage Matrix

	PNO	PNAME	BUDGET	LOC
q1	1	0	1	0
q2	0	1	1	0
q3	0	1	0	1
q4	0	0	1	1

	S1	S2	S3
q1	15	20	10
q2	5	0	0
q3	25	25	25
q4	3	0	0

Attribute Affinity Matrix

	PNO	PNAME	BUDGET	LOC
PNO	45	0	45	0
PNAME	0	80	5	75
BUDGET	45	5	53	3
LOC	0	75	3	78

Step 3 - Clustered Affinity Matrix- Example

Permute rows and columns of the attribute affinity matrix to generate a clustered affinity matrix where attributes in each cluster are in high affinity to each other.

	PNO	PNAME	BUDGET	LOC
PNO	45	0	45	0
PNAME	0	80	5	75
BUDGET	45	5	53	3
LOC	0	75	3	78

	PNO	BUDGET	PNAME	LOC
PNO	45	45	0	0
BUDGET	45	53	5	3
PNAME	0	5	80	75
LOC	0	3	75	78

Bond Energy Algorithm

Step 4- Partitioning - Example

Divide the clustered attributes into non-overlapping partitions such that the number of application queries that access to more than one partition is as small as possible.

	PNO	BUDGET	PNAME	LOC
PNO	TA			
BUDGET				
PNAME			BA	
LOC				

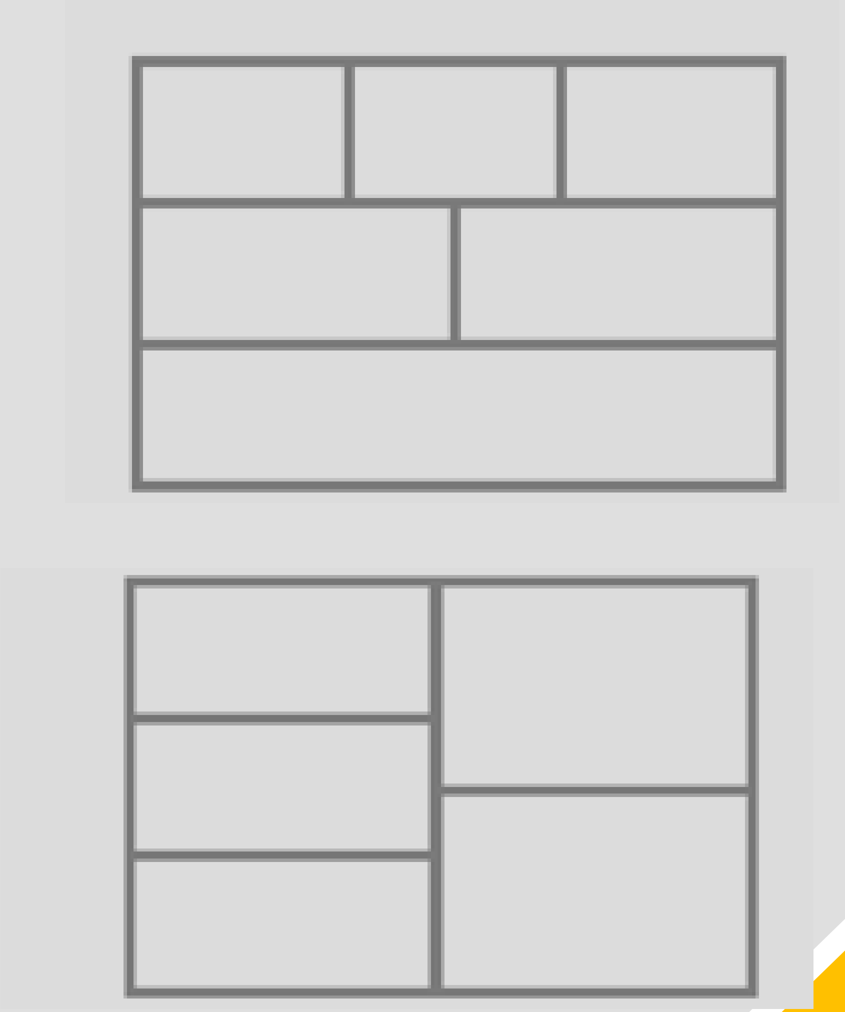
Given:
TQ = set of applications that access only TA
BQ = set of applications that access only BA
OQ = set of applications that access both TA and BA
CTQ = total number of accesses to attributes by TQ
CBQ = total number of accesses to attributes by BQ
COQ = total number of accesses to attributes by OQ

Find:
The point along the diagonal that maximizes

$$CTQ * CBQ - COQ^2$$

Mixed/Hybrid Fragmentation

Consists of a horizontal fragment that is vertically fragmented, or a vertical fragment that is horizontally fragmented.



Mixed/Hybrid Fragmentation

Defined using ***Selection* and *Projection*** operations of relational algebra:

Given a relation R, a mixed fragment is defined as:

$$\sigma_p(\Pi_{a1, \dots, an}(R)) \quad \text{or} \quad \Pi_{a1, \dots, an}(\sigma_p(R))$$

where p is a predicate based on one or more attributes of R and a1.., an are attributes of R.

For example:

$S_1 = \Pi_{\text{staffNo, position, sex, DOB, salary}}(\text{Staff})$

$S_2 = \Pi_{\text{staffNo, fName, lName, branchNo}}(\text{Staff})$

$S_{21} = \sigma_{\text{branchNo='B003'}}(S_2)$

$S_{22} = \sigma_{\text{branchNo='B005'}}(S_2)$

$S_{23} = \sigma_{\text{branchNo='B007'}}(S_2)$

Fragment S_2

StaffNo	fName	lName	BranchNo
SL21	John	White	B005
SG37	Ann	Beech	B003
SG14	David	Ford	B003
SA9	Mary	Howe	B007
SG5	Susan	Brand	B003
SL41	Julie	Lee	B005

Fragment S_{21}

StaffNo	fName	lName	BranchNo
SG37	Ann	Beech	B003
SG14	David	Ford	B003
SG5	Susan	Brand	B003

Fragment S_{22}

StaffNo	fName	lName	BranchNo
SL21	John	White	B005
SL41	Julie	Lee	B005

Fragment S_{23}

StaffNo	fName	lName	BranchNo
SA9	Mary	Howe	B007

Correctness of Fragmentation

Completeness: If relation R is decomposed into fragments R_1, R_2, … R_n, each data item that can be found in R must appear in at least one fragment.

Each attribute in the Staff relation appears in either fragment S_1 or S_2: each (part) tuple appears in fragment S_1 and either fragment S_{21}, S_{22}, or S_{23}.

Reconstruction: Must be possible to define a relational operation that will reconstruct R from the fragments.

The Staff relation can be reconstructed from the fragments using the Natural join operation:

$$S_1 \bowtie (S_{21} \cup S_{22} \cup S_{23}) = \text{Staff}$$

Disjointness: If data item d_i appears in fragment Ri, then it should not appear in any other fragment.

The fragments are disjoint; there can be no staff member who works in more than one branch and S_1 and S_2 are disjoint except for the necessary duplication of primary key.

Fragment S_{21}

StaffNo	fName	lName	BranchNo
SG37	Ann	Beech	B003
SG14	David	Ford	B003
SG5	Susan	Brand	B003

Fragment S_{22}

StaffNo	fName	lName	BranchNo
SL21	John	White	B005
SL41	Julie	Lee	B005

Fragment S_{23}

StaffNo	fName	lName	BranchNo
SA9	Mary	Howe	B007

No Fragmentation

A strategy not to fragment a relation.

The Branch relation contains only a small number of tuples and is not updated very frequently. Rather than trying to horizontally fragment the relation on branch number for example, it would be more sensible to leave the relation whole and simply replicate the Branch relation at each site.

Branch

BranchNo	Street	City	Postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Summary

Distributed Database Design: Fragmentation, Allocation and Replication

Various type of data allocation: Partitioned/ Fragmented, Fully replicated, Partially replicated, Nonreplicated/Non-fragmented.

Comparison of Strategies for Data Allocation based on five important factors.

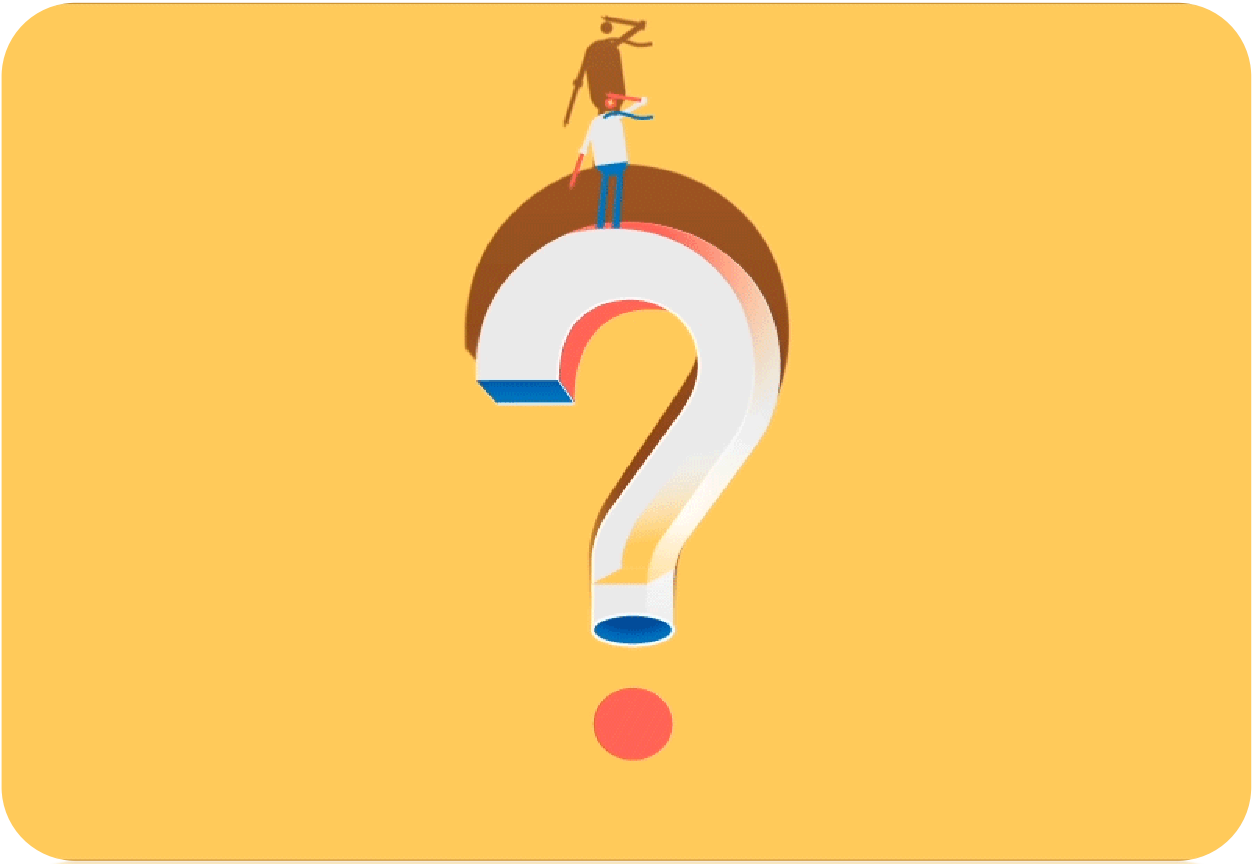
The important of fragmentation and the way of doing fragmentation.

Test your understanding

______ fragmentation allows a user to break a single object into two or more segments, or fragments.

A. Horizontal
B. Vertical
C. Data
D. Request

Any Questions



University of Windsor