Distributed Database System (CS-600)

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**

(Week 15) Lecture 29-30

Objectives: Learning objectives of this lecture are

- **Distributed Database Promises**
- **Distributed Database Design Issues**
- **Fragmentation**
- **Why Fragmentation?**
- **Correctness Rules of Fragmentation**

Reference Material: Database Systems 4th edition Thomas Conolly (Chapter 22)

Distributed Database System (CS-600)

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**

Overview:

In current lecture, we will discuss following points.

- Promises of distributed database system, that must be available at implementation phase.
- What are the design issues occurring at the time of designing a distributed database system?
- What is fragmentation?
- Types of fragmentation?
- Correctness rules for fragmentation.

Recap:

In Previous lecture, we had discuss differences between Distributed Computing and Distributed Databases.

- We discussed distributed database, Distributed Database Management System and Distributed Database System.
- What is centralized databases system and what is distributed database system with examples.
- Data delivery alternatives in different systems.
- What are different types of Delivery modes, types of frequencies and types of communication methods used in data delivery alternatives.

Promises of Distributed Database System:

There are many advantages of DDBS that are categorized as;

1. **Transparent Management of Distributed Data**
2. **Reliability**
3. **Improved Performance**
4. **Easy System Expansion**
5. **Reflects Organizational Structure**
6. **Improved Share ability and local autonomy**
7. **Improved Availability**

Distributed Database System (CS-600)

Mr. Nauman Iqbal email id: nauman@biit.edu.pk Whatsapp# 0321-5821151
Mr. Ahsan email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656

---

1. Transparent Management of Distributed Data:

- Transparency refers to separation of the higher-level semantics of a system from lower-level implementation issues.
- In other words, a transparent system "hides" the implementation details from users.
- The advantage of a fully transparent DBMS is the high level of support that it provides for the development of complex applications.

Example:

Let us start our discussion with an example.

- Consider an engineering firm that has offices in Islamabad, Multan, Lahore and Karachi.
- They run projects at each of these sites and would like to maintain a database of their employees, the projects and other related data.
- Assuming that the database is relational, we can store this information in following relations:
  - EMPLOYEE (ENO, ENAME, TITLE)
  - PROJECT (PNO, PNAME, BUDGET).
  - SALARY (TITLE, AMOUNT)
  - ASIGNMENT (ENO, PNO, RESP, DUR)
- If all of this data were stored in a centralized DBMS, and we wanted to find out the names and employees who worked on a project for more than 12 months, we would specify this using the following SQL query:

```
SELECT ENAME, AMT FROM EMPLOYEE E, ASIGNMENT A, SALARAY S
WHERE A.DUR > 12 AND E.ENO = A.ENO AND S.TITLE = E.TITLE
```

- However, given the distributed nature of this firm's business, it is preferable, under these circumstances, to localize data.
- Such that data about the employees in Multan office are stored in Multan.
- Those in the Lahore office are stored in Lahore, and so forth.
- The same applies to the project and salary information.

Mr. Nauman Iqbal email id: nauman@biit.edu.pk Whatsapp# 0321-5821151
Mr. Ahsan email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656

- Thus, what we are engaged in is a process where we partition each of the relations and store each partition at a different site i.e. fragmentation.
- Furthermore, it may be preferable to duplicate some of this data at other sites for performance and reliability reasons.

Fully transparent access means that the users can still pose the query as specified above, without paying any attention to the fragmentation, location, or replication of data, and let the system worry about resolving these issues. For a system to adequately deal with this type of query over a distributed, fragmented and replicated database, it needs to be able to deal with a number of different types of transparencies that are categorized and described below.

i- **Data Independence**

ii- **Network Transparency**

iii- **Replication Transparency**

iv- **Fragmentation Transparency**

i- Data Independence

- It refers to the immunity of user applications to changes in the definition and organization of data, and vice versa.
- When a user application is written, it should not be concerned with the details of physical data organization.

Data independence can be further categorized into following type of data independencies.

a- Logical Data Independence

Logical data independence refers to the immunity of user applications to changes in the logical structure (i.e., schema) of the database.

b- Physical Data Independence

Logical Physical data independence, on the other hand, deals with hiding the details of the storage structure from user applications.

ii- Network transparencies

- In a distributed database environment, network is also required with data.
- In distributed environment it is preferred that user should be protected from the operational details of the network; possibly even hiding the existence of the network.

- This type of transparency is referred to as network transparency or distribution transparency.

Network transparencies are further categorized as

a- Location Transparency

b- **Naming Transparency**

a- Location Transparency

- Location transparency refers to the fact that the command used to perform a task is independent of both the location of the data and the system on which an operation is carried out.

b- Naming Transparency

- Naming transparency means that a unique name is provided for each object in the database.
- In the absence of naming transparency, users are required to embed the location name (or an identifier) as part of the object name.

iii- Replication Transparency

- It refers to the unawareness of user from replication of data over network.
- In distributed environment which is created by replication; user query can be executed any one replica of the system but user will be unaware of these replicas.

iv- Fragmentation Transparency

- Fragmentation transparency means hiding the detail about fragmentation-based distribution.
- User query may be executed on single or multiple fragmentations.
- The issue is one of finding a query processing strategy based on the fragments rather than the relations, even though the queries are specified on the latter.
- Typically, this requires a translation from what is called a global query to several fragment queries.

Fragmentation transparency is further categorized as

Mr. Nauman Iqbal email id: nauman@biit.edu.pk Whatsapp# 0321-5821151
Mr. Ahsan email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656

---

a- Horizontal Fragmentation Transparency

b- Vertical Fragmentation Transparency

a- Horizontal Fragmentation Transparency

- This type of transparency will be required when data will be distributed on the base of horizontal fragmentation.
- In this scenario user query will be partitioned horizontally.
- User will be unaware of this query partitioning that is refer to the horizontal fragmentation transparency.

b- Vertical Fragmentation Transparency

- This type of transparency will be required when data will be distributed on the base of vertical fragmentation.
- In this scenario user query will be partitioned vertically.
- User will be unaware of this query partitioning that is refer to the vertical fragmentation transparency.

2. Reliability

- Reliability of the system means; system should be trust worthy.
- System should have ability to ensure consistency in database. Distributed DBMSs are intended to improve reliability through transaction.
- A transaction is a series of actions, carried out by a single user or application program, which accesses or changes the contents of the database.

For example, some simple transactions for the ATM might be to WITHDRAW amount from account. To with draw amount series of actions to be required.

- ▪ Check amount within daily limit
- ▪ Check availability of amount
- ▪ Update account’s amount
- ▪ Withdraw amount, etc.

Mr. Nauman Iqbal email id: nauman@biit.edu.pk Whatsapp# 0321-5821151
Mr. Ahsan email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656

- If the transaction fails during execution (before withdraw amount and after update account's amount step), perhaps because of a computer crash, the database will be in an inconsistent state: some changes will have been made and others not.
- Consequently, the changes that have been made will have to be undone to return the database to a consistent state again.

3. Improved Performance

- As the data is located near the site of 'greatest demand', and given the inherent parallelism of distributed DBMSs, speed of database access may be better than that achievable from a remote centralized database.
- Furthermore, since each site handles only a part of the entire database, there may not be the same contention for CPU and I/O services as characterized by a centralized DBMS.

4. Easy System Expansion

- In a distributed environment, it is much easier to accommodate increasing database sizes. Major system overhauls are seldom necessary; expansion can usually be handled by adding processing and storage power to the network.
- Obviously, it may not be possible to obtain a linear increase in "power," since this also depends on the overhead of distribution. However, significant improvements are still possible.

5. Reflects organizational structure

- Many organizations are naturally distributed over several locations.
- For example, DreamHome has many offices in different cities.
- It is natural for databases used in such an application to be distributed over these locations.
- DreamHome may keep a database at each branch office containing details of such things as the staff who work at that location, the properties that are for rent, and the clients who own or wish to rent out these properties.

6. Improved share ability and local autonomy

- The geographical distribution of an organization can be reflected in the distribution of the data; users at one site can access data stored at other sites.
- Data can be placed at the site close to the users who normally use that data.
- In this way, users have local control of the data and they can consequently establish and enforce local policies regarding the use of this data.
- A global database administrator (DBA) is responsible for the entire system.

7. Improved availability

- In a centralized DBMS, a computer failure terminates the operations of the DBMS.
- However, a failure at one site of a DDBMS, or a failure of a communication link making some sites inaccessible, does not make the entire system inoperable.
- Distributed DBMSs are designed to continue to function despite such failures.
- If a single node fails, the system may be able to reroute the failed node's requests to another site.

Disadvantages of DDBMSs:

1. Complexity:

- A distributed DBMS that hides the distributed nature from the user and provides an acceptable level of performance, reliability, and availability is inherently more complex than a centralized DBMS. The fact that data can be replicated also adds an extra level of complexity to the distributed DBMS. If the software does not handle data replication adequately, there will be degradation in availability, reliability, and performance compared with the centralized system, and the advantages we cited above will become disadvantages.

2. Cost:

- Increased complexity means that we can expect the procurement and maintenance costs for a DDBMS to be higher than those for a centralized DBMS. Furthermore, a distributed DBMS requires additional hardware to establish a network between sites. There are ongoing communication costs incurred with the use of this network. There are also additional labor costs to manage and maintain the local DBMSs and the underlying network.

3. **Security:**

   - In a centralized system, access to the data can be easily controlled. However, in a distributed DBMS not only does access to replicated data have to be controlled in multiple locations, but the network itself has to be made secure. In the past, networks were regarded as an insecure communication medium. Although this is still partially true, significant developments have been made to make networks more secure.

4. **Integrity control more difficult:**

   - Database integrity refers to the validity and consistency of stored data. Integrity is usually expressed in terms of constraints, which are consistency rules that the database is not permitted to violate. Enforcing integrity constraints generally requires access to a large amount of data that defines the constraint but which is not involved in the actual update operation itself. In a distributed DBMS, the communication and processing costs that are required to enforce integrity constraints may be prohibitive.

5. **Lack of standards:**

   - Although distributed DBMSs depend on effective communication, we are only now starting to see the appearance of standard communication and data access protocols. This lack of standards has significantly limited the potential of distributed DBMSs. There are also no tools or methodologies to help users convert a centralized DBMS into a distributed DBMS.

Design Issues

There are many following issues that can be raised during building of distributed databases system

1. **Distributed Database Design**
2. **Distributed Directory Management**
3. **Distributed Query Processing**
4. **Distributed Concurrency Control**
5. **Distributed Deadlock Management**
6. **Reliability of DDBMS**
7. **Replication**

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**

---

1. Distributed Database Design

The question that is being addressed is how the database and the applications that run against it should be placed across the sites. There are two basic alternatives to placing data: partitioned (or non-replicated) and replicated. In the partitioned scheme the database is divided into a number of disjoint partitions each of which is placed at a different site. Replicated designs can be either fully replicated (also called fully duplicated) where the entire database is stored at each site, or partially replicated (or partially duplicated) where each partition of the database is stored at more than one site, but not at all the sites. The two fundamental design issues are fragmentation, the separation of the database into partitions called fragments, and distribution, the optimum distribution of fragments.

2. Distributed Directory Management

A directory contains information (such as descriptions and locations) about data items in the database. Problems related to directory management are similar in nature to the database placement problem discussed in the preceding section. A directory may be global to the entire DDBS or local to each site; it can be centralized at one site or distributed over several sites; there can be a single copy or multiple copies.

3. Distributed Query Processing

Query processing deals with designing algorithms that analyze queries and convert them into a series of data manipulation operations. The problem is how to decide on a strategy for executing each query over the network in the most cost-effective way, however cost is defined. The factors to be considered are the distribution of data, communication costs, and lack of sufficient locally-available information. The objective is to optimize where the inherent parallelism is used to improve the performance of executing the transaction, subject to the above-mentioned constraints.

4. Distributed Concurrency Control

Concurrency control involves the synchronization of accesses to the distributed database, such that the integrity of the database is maintained. It is, without any doubt, one of the most extensively studied problems in the DDBS field. The concurrency control problem in a distributed context is somewhat different than in a centralized framework. One not only has to worry about the integrity of a single database, but also about the consistency of multiple copies of the database. The condition that requires all the values of multiple copies of every data item to converge to the same value is called mutual consistency.

5. Distributed Deadlock Management

The deadlock problem in DDBSs is similar in nature to that encountered in operating systems. The competition among users for access to a set of resources (data, in this case) can result in a deadlock if the synchronization mechanism is based on locking. The well-known alternatives of prevention, avoidance, and detection/recovery also apply to DDBSs.

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**

---

6. Reliability of Distributed DBMS

We mentioned earlier that one of the potential advantages of distributed systems is improved reliability and availability. This, however, is not a feature that comes automatically. It is important that mechanisms be provided to ensure the consistency of the database as well as to detect failures and recover from them. The implication for DDBSs is that when a failure occurs and various sites become either inoperable or inaccessible, the databases at the operational sites remain consistent and up to date. Furthermore, when the computer system or network recovers from the failure, the DDBSs should be able to recover and bring the databases at the failed sites up-to-date.

This may be especially difficult in the case of network partitioning, where the sitesare divided into two or more groups with no communication among them.

7. Replication

If the distributed database is (partially or fully) replicated, it is necessary to implement protocols that ensure the consistency of the replicas, i.e. copies of the same data item have the same value. These protocols can be eager in that they force the updates to be applied to all the replicas before the transaction completes, or they may be lazy so that the transaction updates one copy (called the master) from which updates are propagated to the others after the transaction completes.

Fragmentation

Reasons to do Fragmentation

There are following reasons which motivate a distributed database designer to distributed database via fragmentation.

i. Applications views are usually subsets of relation. Therefore, the locality of access of application is defined on not on entire relation but on their subsets. Due to that fragmentation is used to divide a relation into subsets.
ii. Replication needs high volume of memory; while fragmentation requires less memory.
iii. In case of replication; updation, insertion and deletion operation required execution at each site while in case of fragmentation; operation will be performed at related site.
iv. Fragmentation facilitates parallel execution.

Fragmentation Alternatives

There are following two alternative ways to perform fragmentation

- Horizontal
- Vertical

Horizontal Fragmentation

Creating fragments on the base of row is referred as horizontal fragmentation

Example:

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**

Given table

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

Horizontal fragmentation can be performed on the base of budget and conditions are less than 200,000 and greater than equal to 200, 000. Resultant fragments will be as follow.

PROJ_1

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York

PROJ_2

PNO	PNAME	BUDGET	LOC
P3	CAD/CAM	255000	New York
P4	Maintenance	310000	Paris

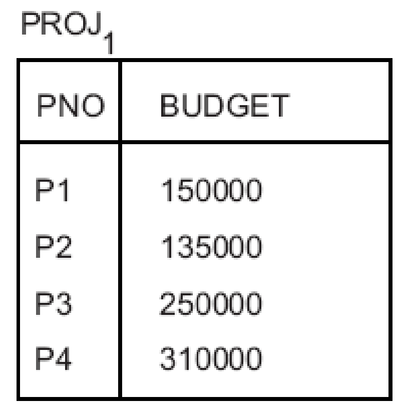
Vertical Fragmentation

Creating fragments on the base of columns is referred as vertical fragmentation.

Example

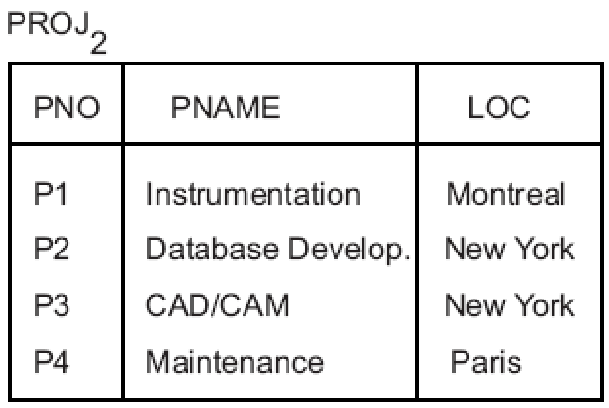
Above table of project can be fragmented as given below, in one fragment there is one column budget and in other PNAME and LOC. In each fragment PNO will be compulsory because it is a key attribute.

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**



$PROJ_1$

PNO	BUDGET
P1	150000
P2	135000
P3	250000
P4	310000



$PROJ_2$

PNO	PNAME	LOC
P1	Instrumentation	Montreal
P2	Database Develop.	New York
P3	CAD/CAM	New York
P4	Maintenance	Paris

Correctness Rules for Fragmentation

Following three rules (called correctness rules) are enforced during fragmentation process.

1- **Completeness**

2- **Reconstruction**

3- **Disjointness**

1- Completeness

- If a table T is fragmented into multiple fragments F1, F2, F3… Fn.
- Then, each record of table T must belong to any of the fragments created.
- If table T has a record R which does not belong to any of the fragments created, then "Completeness" rule is violated.

Example:

Student

RegNo	Name	Age	City
101	Amir	19	ISB
102	Asad	21	RWP
103	Kashif	20	ISB
104	Sadia	25	LHR

Let us create fragments of above Student Table.

Fragment 1 → City = 'ISB'

RegNo	Name	Age	City
101	Amir	19	ISB

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**

103	Kashif	20	ISB

Fragment 2 → City = 'RWP'

RegNo	**Name**	**Age**	**City**
102	Asad	21	RWP

- Here we can see that, Student tale has a record (104, Sadia, 25, LHR) which did not belong to any of the fragments created.

RegNo	**Name**	**Age**	**City**	**Fragment**
101	Amir	19	ISB	F1
102	Asad	21	RWP	F2
103	Kashif	20	ISB	F1
104	Sadia	25	LHR	None

- So 'Completeness' property is violated.
- Now let us create fragments again with some changes.
- **Fragment 1 →** City = 'ISB'

RegNo	**Name**	**Age**	**City**
101	Amir	19	ISB
103	Kashif	20	ISB

- **Fragment 2 →** City != 'ISB'

RegNo	**Name**	**Age**	**City**
102	Asad	21	RWP
104	Sadia	25	LHR

- Here, we can see that all records that belonged to Student table belong to some of the fragment.
- Which means 'Completeness' property of fragmentation is Satisfied.

RegNo	**Name**	**Age**	**City**	**Fragment**
101	Amir	19	ISB	F1
102	Asad	21	RWP	F2
103	Kashif	20	ISB	F1

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**

104	Sadia	25	LHR	F2

NOTE: Fragments must be created in such a way that if new data is inserted in table, rules must be satisfied.

2- Reconstruction

- If a table T is fragmented into multiple fragments F1, F2, F3… Fn.
- No fragments must accept invalid record. Invalid records are those which does not belong to original table T.
- If a fragment accepts a record that is not a member of original table T, then 'Reconstruction' property is said to be 'VIOLATED'

Example:

- **Student table has a constraint that City must be 'Isb' or 'Rwp'. Which means no other city can exist in student table except 'ISB' and 'RWP'.**

Student

RegNo	**Name**	**Age**	**City**
101	Amir	19	ISB
102	Asad	21	RWP
103	Kashif	20	ISB
104	Sadia	25	RWP

Let us create fragments of above Student Table.

- **Fragment 1** → City = 'ISB'

RegNo	**Name**	**Age**	**City**
101	Amir	19	ISB
103	Kashif	20	ISB

- **Fragment 2** → City != 'ISB'

RegNo	**Name**	**Age**	**City**
102	Asad	21	RWP
104	Sadia	25	RWP

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**

- As we can see, all records of Student table have been accommodated in some of fragment. But if another record is inserted (105, Ayesha, 22, LHR) which is an invalid record. Because only those records are valid whose city is either 'ISB' or 'RWP'.
- But above record will be inserted in Fragment 2 because condition is true for it.
- Fragment 2 will become

RegNo	Name	Age	City
102	Asad	21	RWP
104	Sadia	25	RWP
105	Ayesha	22	LHR

- So, Reconstruction Rule is violated because an invalid record exists in Fragment 2.

Correction:

Fragment 1 → City = 'ISB'

RegNo	Name	Age	City
101	Amir	19	ISB
103	Kashif	20	ISB

Fragment 2 → City = 'RWP'

RegNo	Name	Age	City
102	Asad	21	RWP
104	Sadia	25	RWP

- Now if a new record (105, Ayesha, 22, LHR) is inserted, no fragment will accept this invalid record which means 'Reconstruction' Property is satisfied.

3- Disjointness

- If a table T is fragmented into multiple fragments F1, F2, F3… Fn.
- Then, no record R must belong to more than one fragments.
- In other words, each record must belong to one and only one fragment.
- If a record R belongs to more than one fragments, then 'Disjointness' rule will be 'Violated'

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**

Example:

- **Student table has a constraint that City must be 'Isb', 'Rwp' or 'LHR'. Which means no other city can exist in student table except 'ISB' and 'RWP'.**

Student

RegNo	Name	Age	City
101	Amir	19	ISB
102	Asad	21	RWP
103	Kashif	20	ISB
104	Sadia	25	LHR

Let us create fragments of above Student Table.

Fragment 1 → City = 'ISB'

RegNo	Name	Age	City
101	Amir	19	ISB
103	Kashif	20	ISB

Fragment 2 → City = 'RWP'

RegNo	Name	Age	City
102	Asad	21	RWP

Fragment 3 → City != 'ISB'

RegNo	Name	Age	City
102	Asad	21	RWP
104	Sadia	25	LHR

- Here we can see that a record (102, Asad, 21, RWP) belong to Fragment 2 and Fragment 3 which 'Violated' Disjointness rule of Fragmentation.
- As it was said that a record must not belong to more than one fragments.

Distributed Database System (CS-600)

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**

Correction:

Fragment 1 → City = ‘ISB’

RegNo	**Name**	**Age**	**City**
101	Amir	19	ISB
103	Kashif	20	ISB

Fragment 2 → City = ‘RWP’

RegNo	**Name**	**Age**	**City**
102	Asad	21	RWP

Fragment 3 → City = ‘LHR’

RegNo	**Name**	**Age**	**City**
104	Sadia	25	LHR

- Now we can see that, no rules of fragmentation are being violated in above fragments.

x--------------------------x