**Report DBMS**

**Team Details**

1. Meet Sheladiya - 19BIT076
2. Renish Jagani – 19BIT108
3. Rishit Samariya – 19BIT109
4. Rishabh Patel – 19BIT141
5. Ujash Thakkar – 19BIT142
6. Ananya Khandelwal - 19BIT144

**WORK DONE TILL 15/02/2021**

1. File system vs. DBMS

2. 24 Tables of Stadium Seat Booking Management system.

**Assignment – 1**

In Assignment-1 we have stated the limitations of the file system in comparison to DBMS by pointing out the differences in the aspects based on **Data Inconsistency, Data Redundancy, Atomicity Problem, Data Security, Data Integrity and Data Isolation**. Each member described the topic assigned to them and give one example related to the project of each topic.

1. **Data Inconsistency**

|  |  |
| --- | --- |
| **FILE SYSTEM** | **DBMS** |
| **Description-**  Data Inconsistency in file systems means that different files may contain different details of a particular object or person. The retrenchment actually leads to inconsistencies. When the same data is stored in multiple locations, flexibility is possible. | **Description: -**  Inconsistency in databases occurs when the configuration file is marked as such after Ingres encountered a problem while processing the database. This leaves the database inaccessible and unusable until the problem is resolved. Data  Inconsistency is a situation where many tables within data deal with the same data but can find it in different inputs. |
| **Example 1 -** Data Inconsistency occurs in file system when Booking a Seat for stadium if any spectators has record of 2 different phone numbers then system for that data will be unable to identify seat number which is allocated for that spectator, while DBMS would not allow to book the seat. | |
| **Example 2 -** If a person has a book seat for two consecutive scheduled matches and has entered the same data for both matches if any one of the matches gets cancelled then the file system might not give the correct message to that person but in DBMS it has a separate relation between two matches. | |
| **Example 3 -** Data Inconsistency can imply if a spectator orders a meal for two different timings while booking a seat with the same data then the file system would not respond because the file system could not identify which order is cancelled but DBMS database has a unique relation with different input tables. | |
| **Example 4 -** If a person has initially booked a seat for match and entered data like name, phone number and later book the another seat and entered the same data as previous data then in file system it will show error because if any updating happens system would not recognise that in which ticket updating had happened but in DBMS an foreign key is created where the table of one input is related to the other table. | |
| **Example 5 -** While booking seat If any spectator has his home address at two different data storage location it will occur an error in DBMS because if that spectator moved to some other city and then change the address in one of the location but the other location has previous same address so it will lead from data inconsistency to data redundancy. | |
| **Example 6 -** Data Inconsistency constraints can happen if audience from the different location but same street address name books seat file system could not process the further but in DBMS attribute with same value have unique relations and they are bound with different input tables. | |

1. **Data Redundancy**

|  |  |
| --- | --- |
| **FILE SYSTEM** | **DBMS** |
| **Description: –**  In the file system, the same piece of data can occur in multiple tables through various inputs and this situation is known as data redundancy which is highest in the File System. | **Description: –**  In DBMS Data redundancy is lowest, the data is not repeated more than once. |
| **Example 1** – A redundancy situation can occur in file system where two persons  Booked the same seat for the same match, while DBMS won’t allow the booking of the same seat again by the later person. | |
| **Example 2** - Two persons with the same name and surname applied for booking the VIP seats, in the File system it will create redundancy but in DBMS it will bifurcate the two with a unique booking ID. | |
| **Example 3** - File system will create redundancy when more than one person having the same street address will book the tickets, but DBMS will reduce the redundancy by separating streets in one table and then just reference the correct Foreign key to the street table. | |
| **Example 4** – If more than one person chooses to order for the same meal during the match interval, the file system will create redundancy and it will ultimately lead to data corruption i.e., the food staff will not be able to distinguish the orders of the audience, on the other hand DBMS will reduce the redundancy by making separate table for meal and connect the foreign key to the table. | |
| **Example 5** – Data redundancy is bound to take place in file system when all those who opted for a jersey of the same team to wear at the match to cheer up the players and it will lead to decrease in efficiency of work flow of distributing the jerseys while this will not be the case in  DBMS. | |
| **Example 6** – The size of the database will be unnecessarily increased in the File System when the data of the audience from the same city will be stored, while DBMS will segregate the data by making different tables and interconnecting with the help of keys. | |

1. **Atomicity Problems**

|  |  |
| --- | --- |
| **FILE SYSTEM** | **DBMS** |
| **Description: –**  Atomicity is the guarantee that a series of operations either succeed or fail together or refers to “All or nothing”. Which means either all the operations in a transaction executes or none. | **Description: –**  Databases often have features that allow a series of operations to be committed at once as a single transaction there is only success in this. |
| **Example 1** – An Atomicity situation can occur in a file system where during the buying of a ticket if we buy a stadium ticket the amount has been deducted from the account and from the bank's summary account and amount will be credited to the seller. If this all does not happen as an atomic transaction and there is failure during the transaction and there system is not showing the amount either it is received or not, then we will not be able to get a stadium ticket and the amount of ticket will also be reduced. Whether in a DBMS case the transaction will be placed easily and if failure occurs then also their system will show the amount. | |
| **Example 2**- If we pre book the seat with some beverages and snacks it will store in ticket counter file system and it will see by food counter for snacks and beverages for which seats but if the seats are booked and there is failure in system during sharing of data, due to which it will be unable to know food counter which food to whom. It is an atomicity issue in the file system but in DBMS it will be easy to access the data after failure too. | |
| **Example 3** – If the same person book ticket for X, Y, Z for one executive seats and for others VIP lounge the data was store by the ticket counters file system and the ticket are given to person but during the match there is failure in system during sharing of data and there is no data for VIP lounge persons they will be not allowed to seat if there data don’t show it is atomicity problem but in the DBMS during the failure the data will not vanish at all. | |
| **Example 4 –** File system will create atomicity problem during the home delivery of tickets if we book the tickets by phone call and the money has been credited in there account they deliver the ticket by post but there is failure in the system during the ticket seller and post office the data will be not seen by post office because of the failure in file system and ticket will be unable to receive at home but in the DBMS the data will be not erase during the failure and it will easily access by the post office. | |
| **Example 5-** The atomicity problem also occur, during the file system while buying the food during the match with table service the money was paid for table service food but during the failure in system data was lost and money was paid for food with table service but it only show for food not table service but in the DBMS the data will be stored in the server and during failure also it will not erase. | |
| **Example 6-** While buying the supporting team costume it will lead to an atomicity problem in the file system. The amount of costume was paid Rs.1000, and amount was deducted from the buyer account and it will show to the costume counter but during the transaction of money there is the failure in there system it will lead big deal in file system and after the failure the system will not show the payment is done or not or the money has credited in there account or not but the money will be deducted from buyers account. The counter will not give the receipt too. But in DBMS the data will be stored in the server and it will be easily accessed after the failure from the server. And they will show the amount credited and we will get a receipt. | |

1. **Data Security**

|  |  |
| --- | --- |
| **FILE SYSTEM** | **DBMS** |
| **Description** - File systems provide less security than DBMS and also possess the less security enhancing elements as compared to DBMS. | **Description** - DBMS has more security mechanisms as compared to file systems. It contains a range of security controls designed to protect the Database of the system. |
| **Example- 1** To access the database of customers purchasing tickets online we can use security measures in DBMS that include verification while accessing data and allows only authenticated users to access the sensitive database. | |
| **Example-2** DBMS provides features such as access control unlike the File System that includes security mechanisms in a database management system to grant protection against unauthorized access. | |
| **Example-3** Sensitive database of customers trying to purchase the Stadium ticket online can be protected using DBMS. | |
| **Example-4** Datasets such as Seat Number, Payments, and Personal records can be easily backed up and recovered in DBMS in case of data loss. | |
| **Example-5** Unauthorized access of customer’s database by hackers is more prominent in the case of File System than DBMS. | |
| **Example-6** Techniques such as encryption can be used in DBMS to protect sensitive information of ticket buyers. | |

1. **Data Integrity**

|  |  |
| --- | --- |
| **FILE SYSTEM** | **DBMS** |
| **Description: -**  There will be times where it is essential to add certain constraints to the data before entering it into the database. No procedure is provided by the file system to verify these constraints automatically. | **Description: -**  Whereas by imposing user-defined data constraints on its own, DBMS preserves data integrity. |
| **Example 1** — While booking a stadium seat, a user could mistakenly attempt to enter a date field with a phone number. It would prevent the user from making these errors if the device enforces data integrity. | |
| **Example 2** — A person attempts to input a phone number in an incorrect format while entering his personal details for booking a stadium seat. | |
| **Example 3** — In the application of the stadium seat booking system, an error tries to erase wrong records. | |
| **Example 4** — While booking a seat, a user might enter wrong details and attempt to erase a record from a table, but as part of a connection, another table cites the record. | |
| **Example 5** — When there is already a foreign key in a similar table pointing to that number, a user attempts to change the primary key value. | |
| **Example 6** — While entering the date of the match, the user attempts beyond an appropriate range to enter a date. | |

1. **Data Isolation**

|  |  |
| --- | --- |
| **FILE SYSTEM** | **DBMS** |
| **Description: -**  It is often difficult to find information stored in different files in different departments. Sharing information from multiple, separate files is a complex process and usually requires computer programming knowledge. | **Description: -**  The word ‘Isolation’ means separation. In DBMS, Separation is a data domain where no data should affect another and can appear simultaneously. In short, working on one database should start when working on the first database ends. |
| **Example 1** - Data Isolation occurs when a person booking a seat uses a payment system, they actually create a sub-account with the same permissions as everyone else, but they exist differently across the entire network. | |
| **Example 2** - While booking a seat when a person's financial Transaction 2 changes the row, but does not make any changes. Activity 1 and read the unintended details. Now if Transaction 2 reverses its changes already read by Transaction 1 or updates different changes to the database, viewing the data may be incorrect in Transaction 1 records. | |
| **Example 3** - If a person1 while booking seats issues a transaction against at the same time that person2 issues a different transaction, both transactions should operate independently. Databases should either perform person1′s entire transaction before executing person's or vice-versa. This prevents person1’s transaction from reading intermediate data produced as a side effect of part of person's transaction that will not eventually be committed to the database. | |
| **Example 4** - Data Isolation in this situation occurs if a person buys a ticket and books seat and another person tries to book the same seat then the one whose transaction completes first the seat allocated to that person because in DBMS both activities are happening separately. | |
| **Example 5** – while booking a seat if person wants to books VIP seats but in file system doesn’t specify that which one is VIP and which one is Executive lodge and file system books the Executive lodge but in  DBMS specifications between two distinct label seats are mentioned. | |
| **Example 6** – A person wants to book a specific team material but when that person receives other team material then data isolation occurs in the file system but in DBMS data of one table would not affect the data of another table. | |