**Database Systems Assignment 1 Part 1**



**Student’s Name:** Muhammad Sufiyan Baig **Student’s ID:** BDA-24S-001

**Instructions:**

* Answer all the questions clearly and concisely.
* Ensure each answer is well explained, and where necessary, provide examples or diagrams to support your explanation.
* Submit your assignment in PDF format.

**Question 1: Database Management System (DBMS) Overview Tasks:**

1. **Define a Database Management System (DBMS). What are the key functions of a**

**DBMS?**

**Ans**: **Database Management System (DBMS)** refers to a computational system that is used to manage a database. It enables users to store, search for and process data in an organized and time efficient way. DBMS acts as a mediator between users of the database and the database by keeping the datasets well arranged.

**Key Functions of a DBMS:**

* **Data Storage, Retrieval, and Update:** DBMS organizes data logically and also provides easy means for accessing and modifying the data.
* **Data Security:** It affords ways of protecting information that is private from access by other people.
* **Data Integrity:** Finalizes data verification through the use integrity constraints.
* **Concurrency Control:** Coordinates the use of the database among user to ensure that two or more users do not contend for the database.
* **Backup and Recovery:** Performs data backup and offers the means through which to restore information in the event of loss.
* **Data Abstraction:** Offers even different levels of abstraction so as to minimize details from users.

1. **Discuss the advantages of using a DBMS over a traditional file-based system.**

**Ans:** Using a DBMS offers several advantages over traditional file-based systems:

* **Data Redundancy and Inconsistency Control**: DBMS helps to eliminate repeated storage of data and guarantees that all users deals with the current copy of data.
* **Data Integrity:** Some of the benefits of DBMS include Having defined rules that prevent the input of the wrong data into the database.
* **Data Security:** It gives user authentication and role base access control to protect data/information from unauthorized access.
* **Data Independence:** DBMS has data abstraction, which means that one can modify the data structure without altering any of the application layers.
* **Efficient Data Access:** DBMS possesses capability of using complicated query language in the procurement and manipulation of data using efficient search methods.
* **Concurrent Access:** Due to the management of transactions, there is an opportunity for many users to access the database at the same time and there may be some conflict.
* **Backup and Recovery:** Through DBMS, the usual operation of data backup and recovery is simplified to an automated function in the event of a failure.

1. **Briefly explain the three levels of abstraction in a DBMS: physical level, logical level, and view level.**

**Ans.** DBMS uses three levels of abstraction to separate the database structure from the applications accessing it:

1. **Physical Level**: This is the least levels of abstraction and refer to how the data is stored physically in the database. It works with data storage, indexing and disks allocation but these crux aspects are well hidden from the user.

2. **Logical Level**: This level explains what are the data to be stored as well as how the data entities are related. Objectively, it depicts the database in terms of table, fields and relationships but does not make the physical arrangement of storage manifest. Database administrators work at this level.

3. **View Level**: This is the highest level of abstraction and is interested more in how users can be able to access the database. It describes personalized form of the data with respect to the specific user where the user can only see his own data but not the other person’s data.

1. **What is data independence, and why is it important in database systems?**

**Ans:** **Data independence** is the ability to alter the internal definition of a database at a certain tier of the database management system without doing the same at the next higher tier. There are two types of data independence:

* **Physical Data Independence**: Option to swap physical data storage without affecting the logical layout of the same.
* **Logical Data Independence**: The planning flexibility to modify the logical schema without the need to adjust the external view or application programs.

**Importance of Data Independence**:

* **Reduces Maintenance Costs:** It may be adapted at one level, changes can be made at this level of the system without having to rewrite the application programs making the system more flexible.
* **Improved Data Management:** This way resources may be optimized and administered with the system itself but in a manner that does not inflict itself on the users or the usage that they might provide to the developers and administrators of the system.
* **Separation of Concerns:** In different level of abstraction, the developer can just simply concentrate with their task that maybe about how the data is stored or how it will be seen by the user.

**Question 2: Relational Model and Relational Algebra Tasks:**

1. **Define the relational model in the context of databases. What are the key components of a relational database?**

**Ans:** The **relational model** is a model for storing data into structures called **relations** which are like tables. Every given relation is defined as a set of tuples (rows) which contains attributes (columns) as the relation data. This kind of model was proposed by Edgar F. Codd in 1970, and it serves as methodological tools for data definition and data manipulation through declarative means.

**Key Components of a Relational Database**:

* **Relations (Tables)**: The record-based model of the relational model of data, where data is staged in table format in rows and columns.
* **Attributes (Columns)**: Every relation exhibits properties of data that is in the tuple form and have attributes that explain the relation’s features. All the attributes have their unique data type.
* **Tuples (Rows)**: The record field in a table means each row consists of an individual record comprising related data attributes.
* **Keys:**

**.) Primary** **Key**: A name associated with each tuple of a relation in order to avoid occurrence of any two tuples sharing the same value in this particular attribute.

**.) Foreign** **Key**: It is an attribute that connects two relations by having a relation with the primary key of another table.

* **Constraints**: Constraints that were placed at attributes or at tables and these are the likes of unique, reference and not-allow null constraints.

1. **Explain the concept of a primary key and a foreign key. Provide an example of each.**

**Ans:** **Primary** **Key**: In other words, the primary key is the identifier of each record in a table. It makes certain that no two tuples in the relation are similar and also that each of them has its own label. A primary key cannot contain the NULL value.

**Example**: In a Students, there can be primary key at the **student\_id**:

|  |  |  |
| --- | --- | --- |
| **student\_id** | **name** | **age** |
| 1 | Muhammad Sufiyan Baig | 19 |
| 2 | Muhammad Hasan Baig | 17 |
| 3 | Muhammad Anus Baig | 15 |

**Foreign Key**: A foreign key is an attribute belong to one table that refers to the primary key belongs to another table. It creates a dependency between both tables, so that there can be link between them.

**Example**: In an Enrollments table, the ‘**student\_id**’ can in fact be a foreign key to the **student\_id**’ in the ‘Students’ table.

|  |  |  |
| --- | --- | --- |
| **enrollment\_id** | **student\_id** | **course\_name** |
| 101 | 1 | Science |
| 102 | 2 | Math |
| 103 | 3 | History |

In this example, the **student\_id** in the Enrollments table is a foreign key that has a relationship to the primary key of the Students table.

1. **What is relational algebra, and why is it important in database theory? Briefly describe the following operations in relational algebra:**

* **Selection (σ)**
* **Projection(π)**
* **Join (⨝)**
* **Union (∪)**

**Ans: Relational Algebra:**

Relational Algebra is on of the most important languages for manipulating relations (representing a table in a database).

**Importance:**

* Is very useful in query optimization in the database systems.
* Explains the concepts of queries and the results of queries in an understandable way.
* Offers one way of approaching relational database design.

**Key Operations:**

1. **Selection (σ):** Retrieves tuples that meet a condition.

* **Example:** σage>20(Students) selects students older than 20.

1. **Projection (π):** Creates a new relation with specified attributes.

* **Example**: πname, age(Students) retrieves names and ages.

3. **Join (⨝)**: Combines tuples from two relations based on a common attribute.

* **Example**: Students ⋈ Enrollments joins by **student\_id**.

1. **Union (∪)**: Merges two relations, resulting in unique tuples.

* **Example**: StudentsMath ∪ StudentsScience lists all unique students in Math or Science.

**4. How does relational algebra differ from SQL?**

**Ans:** Relational algebra and SQL (Structured Query Language) differ in several ways:

1. **Nature**:

* **Relational Algebra**: This means that it is more of procedural type query language where one gets to state operations on how to fetch data.
* **SQL**: It is a declarative query language, meaning it specifies what data to retrieve without detailing how to retrieve it.

1. **Complexity**:

* **Relational Algebra**: It is a small set of operations (such as selection, projection, join and so on,) which can be arranged to formulate other sophisticated queries.
* **SQL**: It offers a great variety of such built in functions, the functionality of an aggregation and such facility for creating complex query structures.

1. **Use**:

* **Relational** **Algebra**: Originally only used in theoretical exercises and as a basis to understand how the database systems operate.
* **SQL**: Common in practice for querying relational databases, bringing operations more naturally to the user.

1. **Output**:

* **Relational** **Algebra**: Creating tables as output is a requirement, no matter what its input is.
* **SQL**: Can generate different results such as a table, view and even a procedure result which could be in form of insert or update on the data.