**Assignment(1)**

**Class:BCA-II(Sem-III)**

**Subject:Database Management System**

**Topics:**DBMS,advantages of DBMS, Basic Operations of relational algebra

**Submitted To:**

Vibha Gupta

Prof.of Comp.Sci.Department

**Ques1:What is DBMS?Advantages of DBMS?**

**Ans**.The Database Management System (DBMS) is defined as a software system that allows the user to define, create and maintain the database and provide control access to the data.

It is a collection of programs used for managing data and simultaneously it supports different types of users to create, manage, retrieve, update and store information.

Advantages of DBMS

The advantages of the DBMS are explained below −

* **Redundancy problem can be solved.**

In the File System, duplicate data is created in many places because all the programs have their own files which create data redundancy resulting in wastage of memory. In DBMS, all the files are integrated in a single database. So there is no chance of duplicate data.

For example: A student record in a library or examination can contain duplicate values, but when they are converted into a single database, all the duplicate values are removed.

* **Has a very high security level.**

Data security level is high by protecting your precious data from unauthorized access. Only authorized users should have the grant to access the database with the help of credentials.

* **Presence of Data integrity.**

Data integrity makes unification of so many files into a single file. DBMS allows data integrity which makes it easy to decrease data duplicity Data integration and reduces redundancy as well as data inconsistency.

* **Support multiple users.**

DBMS allows multiple users to access the same database at a time without any conflicts.

* **Avoidance of inconsistency.**

DBMS controls data redundancy and also controls data consistency. Data consistency is nothing but if you want to update data in any files then all the files should not be updated again.

In DBMS, data is stored in a single database so data becomes more consistent in comparison to file processing systems.

* **Shared data**

Data can be shared between authorized users of the database in DBMS. All the users have their own right to access the database. Admin has complete access to the database. He has a right to assign users to access the database.

* **Enforcement of standards**

As DBMS have central control of the database. So, a DBA can ensure that all the applications follow some standards such as format of data, document standards etc. These standards help in data migrations or in interchanging the data.

* **Any unauthorized access is restricted**

Unauthorized persons are not allowed to access the database because of security credentials.

* **Provide backup of data**

Data loss is a big problem for all the organizations. In the file system users have to back up the files in regular intervals which lead to waste of time and resources.

DBMS solves this problem of taking backup automatically and recovery of the database.

**Tunability**

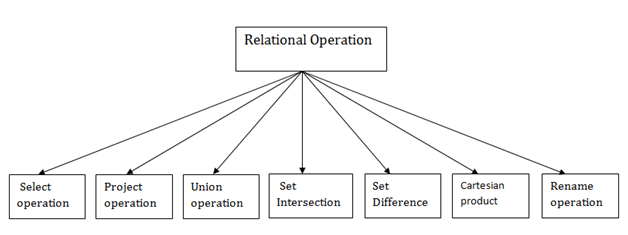
Tuning means adjusting something to get a better performance. Same in the case of DBMS, as it provides tunability to improve performance. DBA adjusts databases to get effective results.

**Ques2:Basic Operations of relational algebra?**

# Ans. Relational Algebra

Relational algebra is a procedural query language. It gives a step by step process to obtain the result of the query. It uses operators to perform queries.

## Types of Relational operation



### 1. Select Operation:

* The select operation selects tuples that satisfy a given predicate.
* It is denoted by sigma (σ).

1. Notation:  σ p(r)

**Where:**

|  |  |  |
| --- | --- | --- |
| **BRANCH\_NAME** | **LOAN\_NO** | **AMOUNT** |
| Downtown | L-17 | 1000 |
| Redwood | L-23 | 2000 |
| Perryride | L-15 | 1500 |
| Downtown | L-14 | 1500 |
| Mianus | L-13 | 500 |
| Roundhill | L-11 | 900 |
| Perryride | L-16 | 1300 |

**σ** is used for selection prediction  
**r** is used for relation  
**p** is used as a propositional logic formula which may use connectors like: AND OR and NOT. These relational can use as relational operators like =, ≠, ≥, <, >, **For example: LOAN Relation**

**Input:**

1. σ BRANCH\_NAME="perryride" (LOAN)

**Output:**

|  |  |  |
| --- | --- | --- |
| **BRANCH\_NAME** | **LOAN\_NO** | **AMOUNT** |
| Perryride | L-15 | 1500 |
| Perryride | L-16 | 1300 |

### 2. Project Operation:

* This operation shows the list of those attributes that we wish to appear in the result. Rest of the attributes are eliminated from the table.
* It is denoted by ∏.

1. Notation: ∏ A1, A2, An (r)

**Where**

**A1**, **A2**, **A3** is used as an attribute name of relation **r**.

**Example: CUSTOMER RELATION**

|  |  |  |
| --- | --- | --- |
| **NAME** | **STREET** | **CITY** |
| Jones | Main | Harrison |
| Smith | North | Rye |
| Hays | Main | Harrison |
| Curry | North | Rye |
| Johnson | Alma | Brooklyn |
| Brooks | Senator | Brooklyn |

**Input:**

1. ∏ NAME, CITY (CUSTOMER)

**Output:**

|  |  |
| --- | --- |
| **NAME** | **CITY** |
| Jones | Harrison |
| Smith | Rye |
| Hays | Harrison |
| Curry | Rye |
| Johnson | Brooklyn |
| Brooks | Brooklyn |

### 3. Union Operation:

* Suppose there are two tuples R and S. The union operation contains all the tuples that are either in R or S or both in R & S.
* It eliminates the duplicate tuples. It is denoted by ∪.

1. Notation: R ∪ S

A union operation must hold the following condition:

* R and S must have the attribute of the same number.
* Duplicate tuples are eliminated automatically.

### Example:

**DEPOSITOR RELATION**

|  |  |
| --- | --- |
| **CUSTOMER\_NAME** | **ACCOUNT\_NO** |
| Johnson | A-101 |
| Smith | A-121 |
| Mayes | A-321 |
| Turner | A-176 |
| Johnson | A-273 |
| Jones | A-472 |
| Lindsay | A-284 |

**BORROW RELATION**

|  |  |
| --- | --- |
| **CUSTOMER\_NAME** | **LOAN\_NO** |
| Jones | L-17 |
| Smith | L-23 |
| Hayes | L-15 |
| Jackson | L-14 |
| Curry | L-93 |
| Smith | L-11 |
| Williams | L-17 |

**Input:**

1. ∏ CUSTOMER\_NAME (BORROW) ∪ ∏ CUSTOMER\_NAME (DEPOSITOR)

**Output:**

|  |
| --- |
| **CUSTOMER\_NAME** |
| Johnson |
| Smith |
| Hayes |
| Turner |
| Jones |
| Lindsay |
| Jackson |
| Curry |
| Williams |
| Mayes |

### 4. Set Intersection:

* Suppose there are two tuples R and S. The set intersection operation contains all tuples that are in both R & S.
* It is denoted by intersection ∩.

1. Notation: R ∩ S

**Example:** Using the above DEPOSITOR table and BORROW table

**Input:**

1. ∏ CUSTOMER\_NAME (BORROW) ∩ ∏ CUSTOMER\_NAME (DEPOSITOR)

**Output:**

|  |
| --- |
| **CUSTOMER\_NAME** |
| Smith |
| Jones |

### 5. Set Difference:

* Suppose there are two tuples R and S. The set intersection operation contains all tuples that are in R but not in S.
* It is denoted by intersection minus (-).

1. Notation: R - S

**Example:** Using the above DEPOSITOR table and BORROW table

**Input:**

1. ∏ CUSTOMER\_NAME (BORROW) - ∏ CUSTOMER\_NAME (DEPOSITOR)

**Output:**

|  |
| --- |
| **CUSTOMER\_NAME** |
| Jackson |
| Hayes |
| Willians |
| Curry |

### 6. Cartesian product

* The Cartesian product is used to combine each row in one table with each row in the other table. It is also known as a cross product.
* It is denoted by X.

1. Notation: E X D

### Example:

**EMPLOYEE**

|  |  |  |
| --- | --- | --- |
| **EMP\_ID** | **EMP\_NAME** | **EMP\_DEPT** |
| 1 | Smith | A |
| 2 | Harry | C |
| 3 | John | B |

**DEPARTMENT**

|  |  |
| --- | --- |
| **DEPT\_NO** | **DEPT\_NAME** |
| A | Marketing |
| B | Sales |
| C | Legal |

**Input:**

1. EMPLOYEE X DEPARTMENT

**Output:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EMP\_ID** | **EMP\_NAME** | **EMP\_DEPT** | **DEPT\_NO** | **DEPT\_NAME** |
| 1 | Smith | A | A | Marketing |
| 1 | Smith | A | B | Sales |
| 1 | Smith | A | C | Legal |
| 2 | Harry | C | A | Marketing |
| 2 | Harry | C | B | Sales |
| 2 | Harry | C | C | Legal |
| 3 | John | B | A | Marketing |
| 3 | John | B | B | Sales |
| 3 | John | B | C | Legal |

### 7. Rename Operation:

The rename operation is used to rename the output relation. It is denoted by **rho** (ρ).

**Example:** We can use the rename operator to rename STUDENT relation to STUDENT1.

1. ρ(STUDENT1, STUDENT)