DBMS notes

Important points

- DBMS provides an organized way of managing, retrieving, and storing a collection of logically related data.
  - Database -> collection of related data that represent some real-world entities
  - Data -> facts that can be recorded like text, numbers, videos, speech, etc
- RDBMS is the same as DBMS but RDBMS has relations b/w the data and the data is stored in tables.
- Types of DBMS →
  1. Relational
  2. Hierarchical => structure similar to tree
  3. Network => similar to hierarchical, this model supports many to many relationships, as child tables can have more than one parent
  4. Object-Oriented => the information is represented as objects, with different types of relationships possible between two or more objects.
- Advantages of DBMS →
  1. DATA INDEPENDENCE -> change the structure of data without changing applications
  2. SHARING OF DATA -> concurrent access to data
  3. INTEGRITY CONSTRAINTS -> check on data before storing
  4. REDUNDANCY CONTROL -> store reference to the data instead of storing a copy of data
  5. PROVIDE BACKUP AND RECOVERY FACILITY
- Different languages in DBMS ->
  1. DDL -> Data Definition Language (CREATE, DROP, ALTER, TRUNCATE, COMMENT, RENAME)
  2. DQL -> Data Query Language (SELECT)

3. DML -> Data Manipulation Language (INSERT, UPDATE, DELETE)
4. DCL -> Data Control Language (GRANT, REVOKE)
5. TCL -> Transaction Controle Language (COMMIT, ROLLBACK, SAVEPOINT)

- NULL value represent a value that is unavailable, 0 is a number, BLANK SPACE is a character THER ARE NOT SAME
- Different levels of abstraction ->
  1. Physical level
  2. Logical level
  3. View level -> describes user interaction with database
- E-R model -> Diagrammatic approach to database design represents entities and relationships between them.
- Different relationships ->
  1. ONE TO ONE
  2. ONE TO MANY / MANY TO ONE
  3. MANY TO MANY
- Concurrency Controle -> Concurrency Control is the management procedure that is required for controlling the concurrent execution of the operations that take place on a database.
- ACID properties in DBMS ->
  1. ATOMICITY -> transactions are completely successful or failed
  2. CONSISTENCY -> the data remains consistent before and after a transaction in a database
  3. ISOLATION -> concurrency control, i.e multiple transactions occur independently without interference
  4. DURABILITY -> changes in successful transactions remain even after system failure occurs
- Normalization -> it is the organization of data to avoid and duplication or redundancy
- Different types of keys in the database ->
  1. Candidate key - A candidate key is an attribute or set of an attribute that can uniquely identify a tuple.
  2. Primary key - A primary key is a column of a table or a set of columns that helps to identify every record present in that table uniquely
  3. Alternate Key - The candidate key other than the primary key is called an alternate key
  4. Foreign Key - Foreign keys are the column of the table which is used to point to the primary key of another table.
- Database partitioning -> process where very large tables are divided into multiple smaller parts. By splitting a large table into smaller, individual tables, queries that access only a fraction of the data can run faster because there is fewer data to scan.
- Difference b/w Unique Key and Primary Key -> U.K. can have NULL value & P.K. can only be one

- 2 tier and 3 tier architecture ->
  - 2 tier architecture -> application program fetches data directly to the database server. e.g -> physically going to a bank the employees check the database directly from their application program. disadvantages -> security, scalability
  - 3 tier architecture -> all the queries are processed at the business layer instead of the database server
- File system vs DBMS ->
  1. data is stored in a remote location and only the needed data can be retrieved instead of getting the whole file
  2. no need for the metadata i.e. where the file is stored, what is the file name, etc.
  3. no protocols for concurrency in the file system but present in DBMS
  4. different users can only access the part of data that is allowed to them
  5. no need to store the same data in different places
- Different normal form ->
  1. 1st Normal Form ->
     It states that an attribute of a table can not hold multiple values. i.e. every column should have at most one value in every row.
  2. 2ns Normal Form ->
     no partial dependency [no part of candidate key should determine non-prime(not a subset of candidate key) attributes] i.e. only candidate key determines the non-prime attributes
  3. 3rd Normal Form ->
     no transitive dependancy[no non-prime attribute should determine a non-prime attribute] i.e. either L.H.S. is candidate key or R.H.S. is a prime-attribute
  4. Boyce Codd Normal Form (BCNF) ->
     L.H.S. of all dependency should be a candidate or super key
- Index ->
  1. implemented using b trees, b+ trees
  2. provides quick lookups based upon queries
  3. needs extra space and can slow down writes
- Types of INDEX →
  - Clustered Index ->
    1. change the way data is stored in the database as per the index
    2. can only be one per table
  - Non-Clustered Index ->
    1. maintain a separate data structure to optimize queries

2. store the reference to the data point separately
3. slower than clustered as we have to go to the data from the points stored separately

- Views -> acts like a new table with only the limited information that we want to share with someone but the data is stored in one place only.

SQL(Structured Query Language)

- Aggregate Functions-> used to summarize data from table
  1. COUNT(COL) - SELECT COUNT(col) as <name> from <table>
  2. SUM(COL)
  3. AVG(COL)
  4. MIN(COL)
  5. MAX(COL)
- Difference b/w DELETE AND TRUNCATE ->
  - DELETE ->

    this command is needed to delete rows from a table based on the condition provided by the WHERE clause.
    1. It deletes only the rows which are specified by the WHERE clause.
    2. It can be rolled back if required.
    3. It maintains a log to lock the row of the table before deleting it and hence it’s slow.
  - TRUNCATE ->
    this command is needed to remove complete data from a table in a database. It is like a DELETE command which has no WHERE clause.
    1. It removes complete data from a table in a database.
    2. It cannot be rolled back even if required.
    3. It doesn’t maintain a log and deletes the whole table at once and hence it’s fast.
- Difference b/w Alter and Update →
  - ALTER →
    - DDL
    - updates structure of the table
  - UPDATE →

        - DML
        - only updates the data of the table
- HAVING VS GROUP BY VS WHERE->
    - GROUP BY -> used to aggregate data with similar values
    - HAVING -> works on the aggregated data from group by
    - WHERE -> works on individual tuple
- Creating an INDEX ->

```
create index name_of_index on table_name(column_name1, coulumn_name2);
```

- Creating a View →

```
create view view_name as select c1, c2 from table_name where condition;
```

- WHERE clause pattern matching ->
    - Wild card patterns ->

      kh% → all strings starting with kh

      % → any string with length >= 0
    - Wild card for specific position ->

      _ → any single character
- FIRST() AND LAST() ->
    - FIRST -> returns the first element in the order in which the select statement would have returned
    - LAST -> returns the last element in the order in which select statement would have returned
- Constraints in SQL ->
    1. UNIQUE
    2. NOT NULL
    3. PRIMARY KEY
    4. DEFAULT
    5. CHECK
    6. FOREIGN KEY
- Domain type in SQL ->
    1. char(n)
    2. varchar(n)
    3. int

4. smallint
5. float(n)

Queries

- Print maximum salary →

```
select max(salary) from emp;
```

- Print name of the employee with MAX SALARY →

```
select name from emp where salary = (select max(salary) from emp);
```

- Print 2nd Highest salary →

```
select max(salary) from emp where salary <> (select max(salary) from emp);
```

```
select salary from emp order by salary offset 1 fetch next 1 rows only;
```

```
select salary from emp order by salary limit 1 offset 1;
```

```
select salary from emp order by salary limit 1 , 1;
```

- Print the name of the employee with 2nd HIGHEST SALARY →

```
select name from emo where salary = ([x]);
```

- Print DEPARTMENT NAME WITH NUMBER OF EMPLOYEES in them →

```
select count(emp) as no_of_emp, dep_name from emp group by dep_name;
```

- Print all departments with LESS THAN 2 EMPLOYEES →

```
select dept from emp group by dept having count(*) < 2;
```

- Print employee's name and department with HIGHEST SALARY DEPARTMENT WISE →

```
select name,dep_name from emp where salary in (select max(salary) from emp group by dep);
```

- Print Nth HIGHEST SALARY →

```
select salary from emp as emp1 where n-1 = (select count(distinct(salary)) from emp where salary > emp1.salary);
```

```
select salary from emp order by salary offset n-1 fetch next 1 rows only;
```

```
select salary from emp order by salary limit 1 offset n-1;
```

```
select salary from emp order by salary limit 1 , n-1;
```

- Print all employees whose names begin with "kh" →

```
select name from emp where name like "kh%";
```

- Print all employees whose name has with "amk" in their name →

```
select name from emp where name like "%amk%";
```

- Print all employees with 3rd CHARACTER in name as "m" →

```
select name from emp where name like "__m%";
```

- Print all employees with names having more than or equal to 4 characters →

```
select name from emp where name like "____%";
```

- Create a table with the exact structure of another table →

```
select * into emp_copy from emp where 3=4;
```