**Q #1) What is** **DBMS used for?**

**Ans:** DBMS, commonly known as Database Management System, is an application system whose main purpose revolves around the **data**. This is a system that allows its users to store the data, define it, retrieve it and update the information about the data inside the database.

**Q #2) What is meant by a Database?**

**Ans:** In simple terms, Database is a collection of **data**in some organized way to facilitate its user’s to easily access, manage and upload the data.

**Q #3) Why is the use of DBMS recommended? Explain by listing some of its major advantages.**

**Ans:** **Some of the major advantages of DBMS are as follows:**

* **Controlled Redundancy:** DBMS supports a mechanism to control redundancy of data inside the database by integrating all the data into a single database and as data is stored at only one place, the duplicity of data does not happen.
* **Data Sharing:**Sharing of data among multiple users simultaneously can also be done in DBMS as the same database will be shared among all the users and by different application programs.
* **Backup and Recovery Facility:**DBMS minimizes the pain of creating the backup of data again and again by providing a feature of ‘backup and recovery’ which automatically creates the data backup and restores the data whenever required.
* **Enforcement of Integrity Constraints:**Integrity Constraints are very important to be enforced on the data so that the refined data after putting some constraints are stored in the database and this is followed by DBMS.
* **Independence of Data:** It simply means that you can change the structure of the data without affecting the structure of any of the application programs.

**Q #4) What is the purpose of normalization in DBMS?**

**Ans:** Normalization is the process of analyzing the relational schemas which are based on their respective functional dependencies and the primary keys in order to fulfill certain properties.

**The properties include:**

* To minimize the redundancy of the Data.
* To minimize the Insert, Delete and Update Anomalies.

**Q #5) What are the different types of languages that are available in the DBMS?**

**Ans:** Basically, there are 3 types of languages in the DBMS as mentioned below:

* **DDL:**DDL is **Data Definition Language** which is used to define the database and schema structure by using some set of SQL Queries like **CREATE**, **ALTER**, **TRUNCATE**, **DROP** and **RENAME.**
* **DCL:**DCL is **Data Control Language** which is used to control the access of the users inside the database by using some set of SQL Queries like **GRANT** and **REVOKE.**
* **DML:**DML is **Data Manipulation Language** which is used to do some manipulations in the database like Insertion, Deletion, etc. by using some set of SQL Queries like **SELECT**, **INSERT**, **DELETE** and **UPDATE.**

**Q #6) What is the purpose of SQL?**

**Ans:** SQL stands for **Structured Query Language** whose main purpose is to interact with the relational databases in the form of inserting and updating/modifying the data in the database.

**Q #7) Explain the concepts of a Primary key and Foreign Key.**

**Ans: Primary Key** is used to **uniquely** identify the records in a database table while **Foreign Key**is mainly used to **link** two or more tables together as this is a particular field(s) in one of the database tables which are the primary key of some other table.

**Example**: There are 2 tables – Employee and Department and both have one common field/column as ‘**ID’** where ID is the **primary** key of the **Employee** table while this is the **foreign** key for the **Department** table.

**Q #8) What are the main differences between Primary key and Unique Key?**

**Ans: Given below are few differences:**

* The **main** difference between the Primary key and Unique key is that the Primary key can never have a **null** value while the Unique key may consist of null value.
* In each table, there can be only one primary key while there can be more than one unique key in a table.

**Q #9) What is the concept of sub-query in terms of SQL?**

**Ans:** Sub-query is basically the query which is included inside some other query and can also be called as an inner query which is found inside the outer query.

**Q #10) What is the use of DROP command and what are the differences between DROP, TRUNCATE and DELETE commands?**

**Ans: DROP** command is a DDL command which is used to drop/delete the existing table, database, index or view from the database.

**The major difference between DROP, TRUNCATE and DELETE commands are:**

**DROP** and **TRUNCATE** commands are the **DDL** Commands which are used to delete tables from the database and once the table gets deleted, all the privileges and indexes that are related to the table also get deleted. These 2 operations **cannot be rolled back** and so should be used with great care.

**DELETE** Command, on the other hand, is a **DML** Command which is also used to delete rows from the table and this **can be rolled back**.

**Note**: It is recommended to use ‘Where’ clause along with the DELETE command else the complete table will get deleted from the database.

**Q #11) What is the main difference between UNION and UNION ALL?**

**Ans:** UNION and UNION ALL are used to join the data from 2 or more tables but **UNION** **removes duplicate rows** and picks the rows which are distinct after combining the data from the tables whereas **UNION ALL** **does not remove the duplicate rows,**it just picks all the data from the tables.

**Q #12) Explain the concept of ACID properties in DBMS?**

**Ans:** ACID properties is the combination of Atomicity, Consistency, Isolation, and Durability properties. These properties are very helpful in allowing a safe and secure way of sharing the data among multiple users.

**Atomicity**: This is based on the concept of “**either all or nothing**” which basically means that if any update occurs inside the database then that update should either be available to all the others beyond user and application program or it should not be available to anyone beyond the user and application program.

**Consistency**: This ensures that the **consistency is maintained in the database** before or after any transaction that takes place inside the database.

**Isolation**: As the name itself suggests, this property states that **each transaction that is occurring is in isolation with others** i.e. a transaction which has started but not yet completed should be in isolation with others so that the other transaction does not get impacted with this transaction.

**Durability**: This property states that the **data should always be in a durable state** i.e. any data which is in the committed state should be available in the same state even if any failure or restart occurs in the system.

**Q #13) What is Correlated Subquery in DBMS?**

**Ans:** A Subquery is also known as a nested query i.e. a query written inside some query. When a Subquery is executed for each of the rows of the outer query then it is termed as a Correlated Subquery.

**Example of Non-Correlated Subquery is:**

Select \* from EMP where ‘RIYA’ IN (Select Name from DEPT where EMP.EMPID=DEPT.EMPID)

Here, inner query is **not executed** for each of the rows of the outer query.

**Q #14) Explain Entity, Entity Type, and Entity Set in DBMS?**

**Ans:** **Entity** is an object, place or thing which has its independent existence in the real world and about which data can be stored in a database. **Eg**: any person, book, etc.

**Entity Type**is a collection of the entities which have the same attributes. **Eg:**STUDENT table contains rows in which each row is an entity holding attributes like name, age , and id of the students, hence STUDENT is an Entity Type which holds the entities having same attributes.

**Entity Set**is a collection of the entities of the same type. **Eg:** A collection of the employees of a firm.

**Q #15) What are the different levels of abstraction in the DBMS?**

**Ans:** There are 3 levels of data abstraction in the DBMS.

**They include:**

* **Physical Level:**This is the lowest level of the data abstraction which states how the data is stored in the database.
* **Logical Level:**This is the next level of the data abstraction which states the type of the data and the relationship among the data that is stored in the database.
* **View Level:** This is the highest level in the data abstraction which shows/states only a part of the database.

**Q #16) What integrity rules exist in the DBMS?**

**Ans:** There are 2 major integrity rules that exist in the DBMS.

**They are:**

* **Entity Integrity:** This states a very important rule that value of a Primary key can never have a NULL value.
* **Referential Integrity:**This rule is related to the Foreign key which states that either the value of a Foreign key is a NULL value or it should be the primary key of any other relation.

**Q #17) What is E-R model in the DBMS?**

**Ans:** E-R model is known as an **Entity-Relationship model** in the DBMS which is based on the concept of the Entities and the relationship that exists among these entities.

**Q #18) What is a functional dependency in the DBMS?**

**Ans:** This is basically a constraint which is useful in describing the relationship among the different attributes in a relation.

**Example:** If there is some relation ‘R1’ which has 2 attributes as Y and Z then the functional dependency among these 2 attributes can be shown as **Y->Z** which states that Z is functionally dependent on Y.

**Q #19) What is 1NF in the DBMS?**

**Ans:** 1NF is known as the **First Normal Form**.

This is the easiest form of the normalization process which states that the **domain** of an attribute should have only atomic values. The objective of this is to remove the duplicate columns that are present in the table.

**Q #20) What is 2NF in the DBMS?**

**Ans:** 2NF is the **Second Normal Form**.

Any table is said to have in the 2NF if it satisfies the following 2 conditions:

* A table is in the 1NF.
* Each non-prime attribute of a table is said to be functionally dependent in totality on the primary key.

**Q #21) What is 3NF in the DBMS?**

**Ans:** 3NF is the **Third Normal Form.**

**Any table is said to have in the 3NF if it satisfies the following 2 conditions:**

* A table is in the 2NFm.
* Each non-prime attribute of a table is said to be non-transitively dependent on every key of the table.

**Q #22) What is BCNF in the DBMS?**

**Ans:** BCNF is the **Boyce Codd Normal Form**which is stricter than the 3NF**.**

**Any table is said to have in the BCNF if it satisfies the following 2 conditions:**

* A table is in the 3NF.
* For each of the functional dependency X->Y that exists, X is the super key of a table.

**Q #23) What is a CLAUSE in terms of SQL?**

**Ans:** This is used with the SQL queries to fetch a specific data as per the requirements on the basis of the conditions that are put in the SQL. This is very helpful in picking the selective records from the complete set of the records.

**Example:** There is a query which has WHERE condition or the query with the HAVING clause.

**Q #24) How can you get the alternate records from the table in the SQL?**

**Ans:** **If you want to fetch the odd numbers then the following query can be used:**

Select EmpId from (Select rowno,EmpId from Emp) where **mod(rowno,2)=1**

If you want to fetch the **even** numbers then the following query can be used:

Select EmpId from (Select rowno,EmpId from Emp) where **mod(rowno,2)=0**

**Q #25) How is the pattern matching done in the SQL?**

**Ans:** With the help of the LIKE operator, pattern matching is possible in the SQL.’**%**’ is used with the Like operator when it matches with the 0 or more characters and **‘\_**’ is used to match the one particular character.

**Example:**

Select \* from Emp where name like ‘b%’

Select \* from Emp where name like ‘hans\_’

**Q #26) What is a join in the SQL?**

**Ans:** A Join is one of the SQL statements which is used to join the data or the rows from 2 or more tables on the basis of a common field/column among them.

**Q #27) What are different types of joins in the SQL?**

**Ans: There are 4 types of SQL Joins:**

* **Inner Join:** This type of join is used to fetch the data among the tables which are common in both the tables.
* **Left Join:** This returns all the rows from the table which is on the left side of the join but only the matching rows from the table which is on the right side of the join.
* **Right Join:** This returns all the rows from the table which is on the right side of the join but only the matching rows from the table which is on the left side of the join.
* **Full Join:** This returns the rows from all the tables on which the join condition has put and the rows which do not match hold null values.

**Q #28) What is meant by trigger?**

**Ans:** Trigger is one of the very important codes or programs which get executed **automatically in response to the events** that occur in a table or a view. **Eg:** If a new record is inserted in an employee database then the data gets created automatically in the related tables like salary, department and roles tables.

**Q #29) Explain Stored Procedure.**

**Ans:** Stored Procedure is a group of SQL statements in the form of a function which has some unique name and is stored in a relational database management systems(RDBMS) and can be accessed whenever required.

**Q #30) What is RDBMS?**

**Ans:** RDBMS is the Relational Database Management System which contains data in the form of the tables and data is accessed on the basis of the common fields among the tables.

**Q #31) What are the different type of relationships in the DBMS?**

**Ans: Relationships in DBMS depicts an association between the tables.**

**Different types of relationships are:**

* **One-to-One:** This basically states that there should be a one-to-one relationship between the tables i.e. there should be one record in both the tables. E**g**: Among a married couple, both wife and husband can have only one spouse.
* **One-to-Many:** This states that there can be many relationships for one i.e. a primary key table hold only one record which can have many, one or none records in the related table. **Eg:** A Mother can have many children.
* **Many-to-Many:**This states that both the tables can be related to many other tables. **Eg:** One can have many siblings and so do they have.

**What are advantages of DBMS over traditional file based systems?**  
**Ans:** Database management systems were developed to handle the following difficulties of typical Fille-processing systems supported by conventional operating systems.  
1. Data redundancy and inconsistency  
2. Difficulty in accessing data  
3. Data isolation – multiple files and formats  
4. Integrity problems  
5. Atomicity of updates  
6. Concurrent access by multiple users  
7. Security problems  
Source: <http://cs.nyu.edu/courses/spring01/G22.2433-001/mod1.2.pdf>

**What are super, primary, candidate and foreign keys?**  
**Ans:**A [superkey](http://en.wikipedia.org/wiki/Superkey" \t "_blank)is a set of attributes of a relation schema upon which all attributes of the schema are functionally dependent. No two rows can have the same value of super key attributes.  
A [Candidate key](http://en.wikipedia.org/wiki/Candidate_key) is minimal superkey, i.e., no proper subset of Candidate key attributes can be a superkey.  
A [Primary Key](http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp?topic=/com.ibm.db2.udb.doc/admin/c0004799.htm) is one of the candidate keys. One of the candidate keys is selected as most important and becomes the primary key. There cannot be more that one primary keys in a table.  
[Foreign key](http://en.wikipedia.org/wiki/Foreign_key) is a field (or collection of fields) in one table that uniquely identifies a row of another table. See [this](http://www.w3schools.com/sql/sql_foreignkey.asp)for an example.

**What is the difference between primary key and unique constraints?**  
**Ans:** Primary key cannot have NULL value, the unique constraints can have NULL values. There is only one primary key in a table, but there can be multiple unique constrains.

**What is database normalization?**  
**Ans:**It is a process of analyzing the given relation schemas based on their functional dependencies and primary keys to achieve the following desirable properties:  
1) Minimizing Redundancy  
2) Minimizing the Insertion, Deletion, And Update Anomalies  
Relation schemas that do not meet the properties are decomposed into smaller relation schemas that could meet desirable properties.  
Source: <http://cs.tsu.edu/ghemri/CS346/ClassNotes/Normalization.pdf>

**What is SQL?**  
SQL is Structured Query Language designed for inserting and modifying in a [relational database system](http://en.wikipedia.org/wiki/Relational_database_management_system).

**What are the differences between DDL, DML and DCL in SQL?  
Ans:** Following are some details of three.  
***DDL***stands for Data Definition Language. SQL queries like CREATE, ALTER, DROP and RENAME come under this.  
***DML***stands for Data Manipulation Language. SQL queries like SELECT, INSERT and UPDATE come under this.  
***DCL*** stands for Data Control Language. SQL queries like GRANT and REVOKE come under this.

**What is the difference between having and where clause?**  
**Ans:** HAVING is used to specify a condition for a group or an aggregate function used in select statement. The WHERE clause selects before grouping. The HAVING clause selects rows after grouping. Unlike HAVING clause, the WHERE clause cannot contain aggregate functions. (See [this](http://newtonapples.com/difference-clause-clause/)for examples).  
See [Having vs Where Clause?](http://quiz.geeksforgeeks.org/having-vs-where-clause/) for more details

**How to print duplicate rows in a table?**  
**Ans:**See <http://quiz.geeksforgeeks.org/how-to-print-duplicate-rows-in-a-table/>

**What is Join?**  
**Ans:** An SQL Join is used to combine data from two or more tables, based on a common field between them. For example, consider the following two tables.

**Table –** Student Table

|  |  |  |
| --- | --- | --- |
| ***ENROLLNO*** | ***STUDENTNAME*** | ***ADDRESS*** |
| 1000 | geek1 | geeksquiz1 |
| 1001 | geek2 | geeksquiz2 |
| 1002 | geek3 | geeksquiz3 |

**Table –** StudentCourse Table

|  |  |
| --- | --- |
| ***COURSEID*** | ***ENROLLNO*** |
| 1 | 1000 |
| 2 | 1000 |
| 3 | 1000 |
| 1 | 1002 |
| 2 | 1003 |

Following is join query that shows names of students enrolled in different courseIDs.

SELECT StudentCourse.CourseID, Student.StudentName

FROM StudentCourse

INNER JOIN Student

ON StudentCourse.EnrollNo = Student.EnrollNo

ORDER BY StudentCourse.CourseID;

The above query would produce following result.

|  |  |
| --- | --- |
| ***COURSEID*** | ***STUDENTNAME*** |
| 1 | geek1 |
| 1 | geek3 |
| 2 | geek1 |
| 3 | geek1 |

**What is Identity?**  
**Ans:** Identity (or AutoNumber) is a column that automatically generates numeric values. A start and increment value can be set, but most DBA leave these at 1. A GUID column also generates numbers; the value of this cannot be controlled. Identity/GUID columns do not need to be indexed.

**What is a view in SQL? How to create one**  
**Ans:** A [view](http://en.wikipedia.org/wiki/View_(SQL))is a virtual table based on the result-set of an SQL statement. We can create using create view syntax.

CREATE VIEW view\_name AS

SELECT column\_name(s)

FROM table\_name

WHERE condition

**What are the uses of view?**  
**1.** Views can represent a subset of the data contained in a table; consequently, a view can limit the degree of exposure of the underlying tables to the outer world: a given user may have permission to query the view, while denied access to the rest of the base table.  
**2.** Views can join and simplify multiple tables into a single virtual table  
**3.** Views can act as aggregated tables, where the database engine aggregates data (sum, average etc.) and presents the calculated results as part of the data  
**4.** Views can hide the complexity of data; for example a view could appear as Sales2000 or Sales2001, transparently partitioning the actual underlying table  
**5.** Views take very little space to store; the database contains only the definition of a view, not a copy of all the data which it presents.  
**6.** Depending on the SQL engine used, views can provide extra security  
Source: [Wiki Page](http://en.wikipedia.org/wiki/View_(SQL))

**What is a Trigger?**  
**Ans:**A [Trigger](http://en.wikipedia.org/wiki/Database_trigger) is a code that associated with insert, update or delete operations. The code is executed automatically whenever the associated query is executed on a table. Triggers can be useful to maintain integrity in database.

**What is a stored procedure?**  
**Ans:** A [stored procedure](http://en.wikipedia.org/wiki/Stored_procedure) is like a function that contains a set of operations compiled together. It contains a set of operations that are commonly used in an application to do some common database tasks.

**What is the difference between Trigger and Stored Procedure?**  
**Ans:** Unlike Stored Procedures, Triggers cannot be called directly. They can only be associated with queries.

**What is a transaction? What are ACID properties?**  
**Ans:** A [Database Transaction](http://en.wikipedia.org/wiki/Database_transaction) is a set of database operations that must be treated as whole, means either all operations are executed or none of them.  
An example can be bank transaction from one account to another account. Either both debit and credit operations must be executed or none of them.  
[ACID](http://en.wikipedia.org/wiki/ACID)(Atomicity, Consistency, Isolation, Durability) is a set of properties that guarantee that database transactions are processed reliably.

**What are indexes?**  
**Ans:** A [database index](http://en.wikipedia.org/wiki/Database_index) is a data structure that improves the speed of data retrieval operations on a database table at the cost of additional writes and the use of more storage space to maintain the extra copy of data.  
Data can be stored only in one order on disk. To support faster access according to different values, faster search like binary search for different values is desired, For this purpose, indexes are created on tables. These indexes need extra space on disk, but they allow faster search according to different frequently searched values.

**What are clustered and non-clustered Indexes?**  
**Ans:** Clustered indexes is the index according to which data is physically stored on disk. Therefore, only one clustered index can be created on a given database table.  
Non-clustered indexes don’t define physical ordering of data, but logical ordering. Typically, a tree is created whose leaf point to disk records. [B-Tree](http://en.wikipedia.org/wiki/B-tree) or [B+ tree](http://en.wikipedia.org/wiki/B+_tree) are used for this purpose.

**Q. There is a table where only one row is fully repeated. Write a Query to find the Repeated row**

| **Name** | **Section** |
| --- | --- |
| abc | CS1 |
| bcd | CS2 |
| abc | CS1 |

In the above table, we can find duplicate row using below query.

SELECT name, section FROM tbl

GROUP BY name, section

HAVING COUNT(\*) > 1

**Q. Query to find 2nd highest salary of an employee?**

SELECT max(salary) FROM EMPLOYEES WHERE salary IN

(SELECT salary FROM EMPLOYEEs MINUS SELECT max(salary)

FROM EMPLOYEES);

OR

SELECT max(salary) FROM EMPLOYEES WHERE

salary <> (SELECT max(salary) FROM EMPLOYEES);

**Q.Consider the following Employee table. How many rows are there in the result of following query?**

**ID   salary   DeptName**  
1    10000      EC  
2    40000      EC  
3    30000      CS  
4    40000      ME  
5    50000      ME  
6    60000      ME  
7    70000      CS

**How many rows are there in the result of following query?**

SELECT E.ID

FROM  Employee E

WHERE  EXISTS  (SELECT E2.salary

FROM Employee E2

WHERE E2.DeptName = 'CS'

AND   E.salary > E2.salary)

Following 5 rows will be result of query as 3000 is the minimum salary of CS Employees and all these rows are greater than 30000.

2  
4  
5  
6  
7

**Q. Write a trigger to update Emp table such that, If an updation is done in Dep table then salary of all employees of that department should be incremented by some amount (updation)**

Assuming Table name are Dept and Emp, trigger can be written as –

CREATE OR REPLACE TRIGGER update\_trig

AFTER UPDATE ON Dept

FOR EACH ROW

DECLARE

CURSOR emp\_cur IS SELECT \* FROM Emp;

BEGIN

FOR i IN emp\_cur LOOP

IF i.dept\_no = :NEW.dept\_no THEN

DBMS\_OUTPUT.PUT\_LINE(i.emp\_no);  --  for printing those

UPDATE Emp                      -- emp number which are

SET sal = i.sal + 100           -- updated

WHERE emp\_no = i.emp\_no;

END IF;

END LOOP;

END;

**Q. There is a table which contains two column Student and Marks, you need to find all the students, whose marks are greater than average marks i.e. list of above average students.**

SELECT student, marks

FROM table

WHERE marks > SELECT AVG(marks) from table;

**Q.Name the student who has secured third highest marks using sub queries.**

SELECT Emp1.Name

FROM Employee Emp1

WHERE 2 = (SELECT COUNT(DISTINCT(Emp2.Salary))

FROM Employee Emp2

WHERE Emp2.Salary > Emp1.Salary

)

\*LOGIC- Number of people with salary higher than this person will be 2.

**Q. Why we cannot use WHERE clause with aggregate functions like HAVING ?**

The difference between the having and where clause in SQL is that the where clause canNOT be used with aggregates, but the having clause can. Please note : It is not a predefined rule but by and large you’ll see that in a good number of the SQL queries, we use WHERE prior to GROUP BY and HAVING after GROUP BY.

The Where clause acts as a pre filter where as Having as a post filter.

The where clause works on row’s data, not on aggregated data.

Let us consider below table ‘Marks’.

Student       Course      Score

a                c1             40  
a                c2             50  
b                c3             60  
d                c1             70  
e                c2             80

Consider the query

SELECT Student, sum(Score) AS total

FROM Marks

This would select data row by row basis. The having clause works on aggregated data.

For example,  output of below query

SELECT Student, sum(score) AS total FROM Marks

Student     Total  
a             90  
b             60  
d             70  
e             80

When we apply having in above query, we get

SELECT Student, sum(score) AS total

FROM Marks having total > 70

Student     Total  
a             90  
e 80

**Q. Difference between primary key and unique key and why one should use unique key if it allows only one null ?**

Primary key:

* Only one in a row(tuple).
* Never allows null value(only key field).
* Unique key identifier and can not be null and must be unique.

Unique Key:

* Can be more than one unique key in one row.
* Unique key can have null values(only single null is allowed).
* It can be a candidate key.
* Unique key can be null and may not be unique.

**Q. What’s the difference between materialized and dynamic view?**

Materialized views

* Disk based and are updated periodically based upon the query definition.
* A materialized table is created or updated infrequently and it must be synchronized with its associated base tables.

Dynamic views

* Virtual only and run the query definition each time they are accessed.
* A dynamic view may be created every time that a specific view is requested by the user.

**Q. What is embedded and dynamic SQL?**

Static or Embedded SQL

* SQL statements in an application that do not change at runtime and, therefore, can be hard-coded into the application.

Dynamic SQL

* SQL statements that are constructed at runtime; for example, the application may allow users to enter their own queries.
* Dynamic SQL is a programming technique that enables you to buildSQL statements dynamically at runtime. You can create more general purpose, flexible applications by using dynamic SQL because the full text of a SQL statement may be unknown at compilation.

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Static (embedded) SQL** | **Dynamic (interactive) SQL** |
| 1. | In static SQL how database will be accessed is predetermined in the embedded SQL statement. | In dynamic SQL, how database will be accessed is determined at run time. |
| 2. | It is more swift and efficient. | It is less swift and efficient. |
| 3. | SQL statements are compiled at compile time. | SQL statements are compiled at run time. |
| 4. | Parsing, validation, optimization, and generation of application plan are done at compile time. | Parsing, validation, optimization, and generation of application plan are done at run time. |
| 5. | It is generally used for situations where data is distributed uniformly. | It is generally used for situations where data is distributed non-uniformly. |
| 6. | EXECUTE IMMEDIATE, EXECUTE and PREPARE statements are not used. | EXECUTE IMMEDIATE, EXECUTE and PREPARE statements are used. |
| 7. | It is less flexible. | It is more flexible. |

http://docs.oracle.com/cd/A87860\_01/doc/appdev.817/a76939/adg09dyn.htm

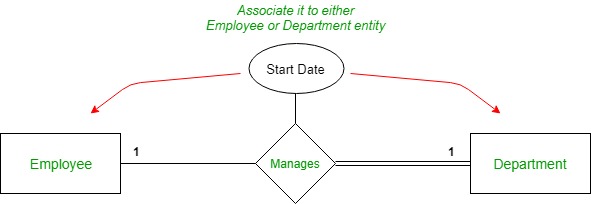
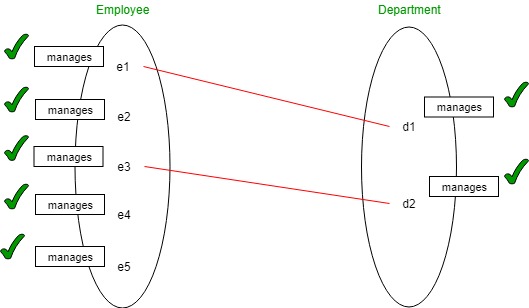
**Q. What is the difference between CHAR and VARCHAR?**

* CHAR and VARCHAR are differ in storage and retrieval.
* CHAR column length is fixed while VARCHAR length is variable.
* The maximum no. of character CHAR data type can hold is 255 character while VARCHAR can hold up to 4000 character.
* CHAR is 50% faster than VARCHAR.
* CHAR uses static memory allocation while VARCHAR uses dynamic memory allocation.

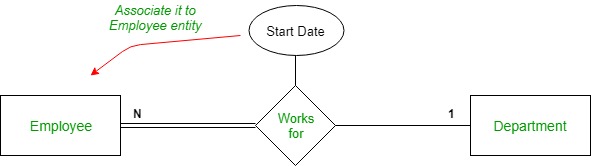
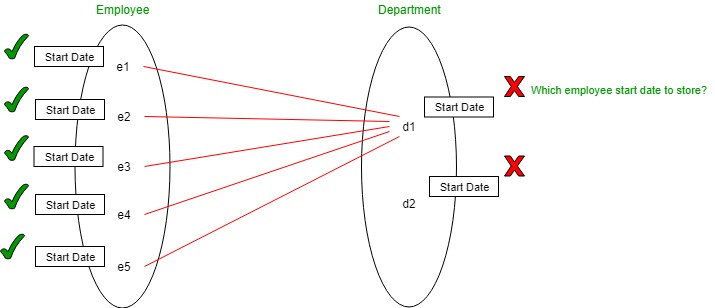
DBMS | Attributes to Relationships in ER Model

In ER model, entities have attributes which can be of various types like single-valued, multi-valued, composite, simple, stored, derived and complex. But relationships can also have attributes associated to them. Generally it is not recommended to give attributes to the relationships if not required because while converting the ER model into Relational model, things may get complex and we may require to create a separate table for representing the relationship. Let us see various cases and when we need to give attributes to the relationship with the help of examples:

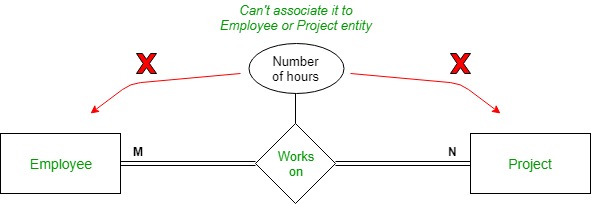
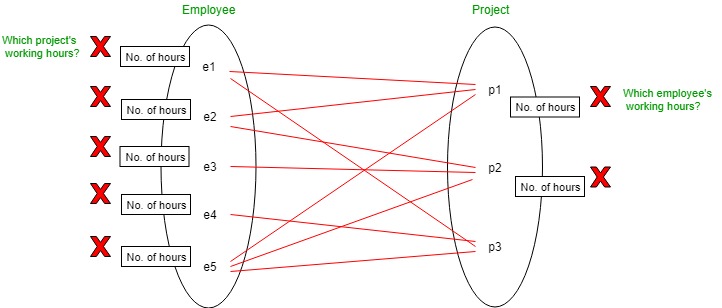
**1. One to one relationship:**  
In an organisation an employee manages a department and each department is managed by some employee. So, there is a total participation of em>Department entity and there is *one to one* relationship between the given entities. Now, if we want to store the *Start\_Date*from which the employee started managing the department then we may think that we can give the *Start\_Date*attribute to the relationship *manages*. But, in this case we may avoid it by associating the *Start\_Date*attribute to either *Employee*or *Department*entity.

**2. One to many relationship:**  
In an organisation many employees can work for a department but each employee can work for only a single department. So, there is a *one to many* relationship between the entities. Now if we want to store the *Start\_Date*when employee started working for the department, then instead of assigning it to the relationship we should assign it to the *Employee*entity. Assigning it to the *employee*entity makes sense as each employee can work for only single department but on the other hand one department can have many employees working under it and hence, it wouldn’t make sense if we assign Start\_Date attribute to Department.

**3. Many to many relationship:**  
In an organisation an employee can work on many projects simultaneously and each project can have many employees working on it. Hence, it’s a *many to many* relationship. So here assigning the *Number\_of\_Working\_hours*to the employee will not work as the question will be that it will store which project’s working hours because a single employee can work on multiple projects. Similar the case with the *project*entity. Hence, we are forced to assign the *Number\_of\_Working\_hours*attribute to the relationship.

**Conclusion:** Give attributes to a relationship only in the case of **many to many** relationship.

# Difference between RDBMS and OODBMS

**RDBMS:**  
RDBMS stands for Relational Database Management System. It is a database management system based on the relational model i.e. the data and relationships are represented by a collection of inter-related tables. It is a DBMS that enables the user to create, update, administer and interact with a relational database. RDBMS is the basis for SQL, and for all modern database systems like MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access.

**OODBMS:**  
OODBMS stands for Object-Oriented Database Management System. It is a DBMS where data is represented in the form of objects, as used in object-oriented programming. OODB implements object-oriented concepts such as classes of objects, object identity, polymorphism, encapsulation, and inheritance. An object-oriented database stores complex data as compared to relational database. Some examples of OODBMS are Versant Object Database, Objectivity/DB, ObjectStore, Caché and ZODB.

**Difference Between RDBMS and OODBMS:**

| **BASIS** | **RDBMS** | **OODBMS** |
| --- | --- | --- |
| Long Form | Stands for Relational Database Management System. | Stands for Object Orientedl Database Management System. |
| Way of storing data | Stores data in Entities, defined as tables hold specific information. | Stores data as Objects. |
| Data Complexity | Handles comparitively simpler data. | Handles larger and complex data than RDBMS. |
| Grouping | Entity type refers to the collection of entity that share a common definition. | Class describes a group of objects that have common relationships, behaviors, and also have similar properties. |
| Data Handeling | RDBMS stores only data. | Stores data as well as methods to use it. |
| Main Objective | Data Independece from application program. | Data Encapsulation. |
| Key | A Primary key distinctively identifies an object in a table.. | An object identifier (OID) is an unambiguous, long-term name for any type of object or entity. |

Disadvantages of DBMS

There are [many advantages](https://www.geeksforgeeks.org/bbms-advantages-dbms-file-system/) and disadvantages of DBMS (Database Management System). Disadvantages of DBMS are explained as following below.

**1. Increased Cost:**  
These are different types of costs:

1. **Cost of Hardware and Software –**  
   This is the first disadvantage of database management system. This is because for DBMS, it is mandatory to have a high speed processor and also a large memory size because now a days there is a large amount of data in every field which needs to be store safely and with a security.

The requirement of these large amount of space and a high speed processor needs an expensive hardware and also an expensive software too. That is there is a requirement of sophisticated hardware and software which means that we need to upgrade the hardware which is used for file-based system. Hardware and Software, both requires maintenance which costs very high. All the operating, Training (all levels including programming, application development, and database administration), licensing, and regulation compliance costs very high.

1. **Cost of Staff Training –**  
   Educated staff (database administrator, application programmers, data entry operations) who maintains the database management system also requires good amount. We need the database system designers to be hired along with application programmers. Alternatively the services of some software house need to be taken. So there is a lot of money which needs to be spent for developing software.
2. **Cost of Data Conversion –**  
   We need to convert our data into database management system, there is a requirement of lot of money as it adds on to the cost of the database management system. This is because, for this conversion we need to hire database system designers whom we have to pay a lot of money and also services of some software house will be required. All this shows that a high initial investment for hardware, software and trained staff is required by DBMS. So, altogether Database Management System results in a costlier system.

**2. Complexity:**  
As we all know that now a days all companies are using the database management system as it fulfils lots of requirement and also solves the problem. But a problem arises, that is all these functionality has made database management system an extremely complex software. For the proper requirement of DBMS it is very important to have a good knowledge of it by the developers, DBA, designers and also the end users. This is because if any one of them do not acquire a proper and complete skills than this may lead to data loss or database failure.

These failure may lead to bad design decisions due to which there may be a serious and bad consequences for the organization. So this complex system needs to be understood by everyone using it. As it cannot be managed very easily. All this shows that database management system is not a child’s game as it cannot be managed very easily. It requires a lot of management. A good staff is needed to manage this database at the times when it becomes very complicated to decide where to pick data from and where to save it.

**3. Currency Maintenance:**  
This is very necessary to keep your system current because efficiency which is one of the biggest factor and need to be overlook must be maximised. That is we need to maximise the efficiency of the database system to keep our system current. For this, frequent updation must be performed on all the components as new threats comes daily. DBMS should be updated according to the current scenario. Also security measures must be needed. Due to advancement in database technology, training cost tends to be significant.

**4. Performance:**  
Traditional file system is written for small organizations and for some specific applications due to which performance is generally very good. But for the small scale firms, DBMS does not give a good performance as its speed is very slow. As a result some applications will not run as fast as they could. Hence it is not good to use DBMS for the small firms. Because performance is a factor which is overlooked by everyone. If performance is good than everyone (developers, designers, end users) will use it easily and it will be user friendly too as speed of the system totally depends on the performance and performance needs to be good.

**5. Frequency Upgrade/Replacement Cycles:**  
Now a days in this world we need to stay up-to-date about the latest technologies, developments arriving in the market. Frequent upgrade of the products is done by the DBMS vendors in order to add new functionality to the systems. New upgrade versions of the software often come bundled. Sometimes these updates also need hardware upgrades. Sometimes these changes and updating are so fast that the users don’t feel it difficult to work with that system because it is not easy to learn new commands and understanding them again and again when the new upgrades are done. All these upgrades also costs money in order to train users, designers etc to use the new features.