Interview questions

Ques-1) What is a Database? How is it different from DBMS or RDBMS ?

Ans - The database is an organized collection of structured data to make it easily accessible, manageable and updated. In simple words, you can say, a database in a place where the data is stored. The best analogy is the library. The library contains a huge collection of books of different genres, here the library is a database and books are the data.

Database Management System (DBMS) is a software that is used to define, create and maintain a database and provides controlled access to the data,or basically it's a software that interacts with the database. and RDBMS can be considered as a specific form of the DBMS , in which data is divided into different tables and these tables are further being related to each other by some key columns.

Ques-2) What is SQL ? How is it different from Mysql, oracle, Mariadb or sqlite?

Ans - SQL stands for Structured Query Language. SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems. SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database.

Whereas Mysql, oracle or sqlite are some common RDBMS that works on the concept of SQL (so basically SQL is a Standard and Mysql or oracle are the software which follows those standard.)

There are some basic differences about all of these db's which can be found here:

https://towardsdatascience.com/top-10-databases-to-use-in-2021-d7e6a85402ba

Ques-3)What are the different types of Statement in SQL?

There are five types of SQL commands: DDL, DML, DCL, TCL, and DQL. (There are books which also refer to 4 types – DDL, DML, DCL, TCL.)

<u>1. Data Definition Language (DDL)</u>

DDL changes the structure of the table like creating a table, deleting a table, altering a table, so whenever you are actually changing the structure of a table by a particular command, that will come under DDL.

All the commands of DDL are auto-committed, which means it permanently saves all the changes in the database.

Here are some commands that come under DDL:

1 CREATE
2 ALTER
3 DROP
4 TRUNCATE

2. Data Manipulation Language (DML)

- DML commands are used to modify the database. It is responsible for all forms of changes in the database. So whenever you are making any kind of change to a database without modifying the original structure of it.
- The command of DML is not auto-committed, which means it can't permanently save all the changes in the database. They can be rolled back.

Here are some commands that come under DML:

1 INSERT
2 UPDATE
3 DELETE

3- Data Control Language(DCL)

DCL commands are used to grant and take back authority from any database user to view or edit the database or a particular instance of it

Here are some commands that come under DCL:

1 Grant
2 Revoke

4- Transaction Control Language(TCL)

TCL commands can only be used with DML commands like INSERT, DELETE and UPDATE only.

These operations are automatically committed in the database that's why they cannot be used while creating tables or dropping them.

Here are some commands that come under TCL:

1 COMMIT
2 ROLLBACK

3 SAVEPOINT

5- Data Query Language(DQL)

DQL is used to fetch the data from the database.

It uses only one command:

- SELECT

Ques4- What are the different constraints we can use in Mysql?

SQL constraints are a set of rules implemented on tables in relational databases to dictate what data can be inserted, updated or deleted in its tables. This is done to ensure the accuracy and the reliability of information stored in the table. Constraints enforce limits to the data or type of data that can be inserted/updated/deleted from a table. The purpose of constraints is to maintain the data integrity during an update/delete/insert into a table. Once the constraint is placed, if any operation in the database does not follow the rules specified by the constraint, the particular operation is aborted.
there are few constraint that we can have -

1. **UNIQUE KEY** - The UNIQUE constraint specifies that no cell value in a column can be repeated throughout the table. It can have NULL or missing values in the data

2. **NOT NULL** - A NOT NULL constraint specifies that no cell value for any row in this column can be blank. Generally, this rule is applied to columns that capture information that is absolutely vital to identify and extract data from a table.

3. **PRIMARY KEY** - (Unique + NOT NULL) PRIMARY KEYS are unique identifiers for each row present in a table. They can be values present in a single column of a table or a combination of multiple columns in the table. The PRIMARY KEY column cannot be NULL and has to be UNIQUE.

4. **DEFAULT** - The DEFAULT constraint is used to specify a default value that is to be entered in any record in a particular column is left blank. The default value will be added to all new records if no other value is specified.

5. **CHECK Constraint** - The CHECK constraint is used to ensure that all the records in a certain column follow a specific rule. Generally, this constraint is used to enforce business logic on values in a column to make sure that no

corrupt information is entered. For example, let's say I am working with a table where if any candidate is of age less than 18, then his data should not be saved into the table, so to enforce this logic we will be using a check constraint.

6 **Foreign Key** - The foreign key constraint is used to prevent operations in a relational database that would destroy links between tables. The FOREIGN KEY is a column (or a group of columns) in one table, that refers to the PRIMARY KEY of another table. The table with the FOREIGN KEY is called the child table while the referenced table with the PRIMARY KEY is called the parent table. so it basically used to maintain the referential integrity of the table

Ques-5) What are the different datatypes available in mysql, and when to use which ?Difference between VARCHAR V/S CHAR, DECIMAL V/S Float or Double , Timestamp V/S Datetime

Ans - There are so many data types in Mysql, some of them are as follows-

1. CHAR(n)
2. VARCHAR(n)
3. INT
4. DECIMAL(precision, scale)
5. FIOAT(p)
6. DOUBLE(p)
7. DATE
8. DATETIME
9. TIMESTAMP
10. ENUM

CHAR V/S VARCHAR

CHAR

1. Used to store character string value of **fixed length**.
2. The maximum no. of characters the data type can hold is **255 characters**.
3. It's **50% faster** than VARCHAR.
4. Uses **static memory allocation**.

VARCHAR

1. Used to store **variable length** alphanumeric data.
2. The maximum this data type can hold is up to
   - Pre-MySQL 5.0.3: **255 characters**.
   - Post-MySQL 5.0.3: **65,535 characters** shared for the row.
3. It's **slower** than CHAR.
4. Uses **dynamic memory allocation**.

Apart from the that, VARCHAR(n) has an overhead of 2 bytes compared to char(n) , so if we know all the data will be of the same length , VARCHAR(n) will take more space.

DECIMAL(P,S) V/S FLOAT or double

The decimal data type is a fixed point data type and the calculations are exact, whereas the float and the double datatype are floating point types and the calculations are approximate.

Decimal takes more space, but it is accurate . whereas float and double uses less memory to accommodate large numbers but it comes at the cost of precision.

Datetime V/S Timestamp

The DATETIME type is used when you need values that contain both date and time information. MySQL retrieves and displays DATETIME values in 'YYYY-MM-DD HH:MM:SS' format. The supported range is '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.

The TIMESTAMP data type has a range of '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC. It has varying properties, depending on the MySQL version and the SQL mode the server is running in. Timestamp are actually used when we try to create meta data like when something was updated.

You can learn more about all the other datatype available here:
https://dev.mysql.com/doc/refman/8.0/en/data-types.html

Ques6) What is Enum in SQL? and why is it used?

An ENUM is a string object with a value chosen from a list of permitted values that are enumerated explicitly in the column specification at table creation time. ENUM provides us more control on the way we can use string literals if there are only few specific value that can be entered into the column.

For example if you are making a column to store a person's gender .It can only take a few possible values into the column, if any other value is being tried to store into that column it will directly throw an error, so it can be a good way to ensure the data quality.

You can learn more about it here:

https://codingsight.com/what-is-mysql-enum-top-12-key-facts-you-need-to-know/

Ques-7) What are ACID properties in Mysql?

ACID (Atomicity, Consistency, Isolation, Durability) is a set of properties of database transactions intended to guarantee validity even in the event of errors, power failures, etc.

In the context of databases, a sequence of database operations that satisfies the ACID properties, and thus can be perceived as a single logical operation on the data, is called a transaction.For example, a transfer of funds from one bank account to another involves debiting from one account and crediting to another, and this whole process is a single transaction.

Atomicity

All statements of a transaction must succeed completely, or fail completely in each and every situation, including power failures, errors and crashes. Example - Debiting and crediting in a money transfer transaction, both must happen either together or not at all.

Consistency

The database must remain in a consistent state after any transaction. Data in the database should not have any changes other than intended after the transaction completion.

Isolation

Isolation ensures that concurrent execution of transactions leaves the database in the same state that would have been obtained if the transactions were executed sequentially.

Durability

Durability guarantees that once a transaction has been committed, it will remain committed even in the case of a system failure which actually means recording the completed transactions (or their effects) in non-volatile memory.

Ques8 - How SQL is different From Excel or any file based storage System?

SQL and spreadsheet applications such as Microsoft Excel are different things. They all indeed work with data in tables or structured data. Still, SQL is a computer language we use to communicate with relational databases through another set of software called relational database management systems (RDBMS) such as MySQL, Excel is an application that works like a very smart calculator that can do many different calculating and analysis tasks.

Apart from that there are many other working differences –

- At its full capacity, Excel can store about 1 million rows and 16,000 columns of data. Those numbers are relatively small compared to what a database system can store.
- We can indeed store up to about 1 million rows of data in an Excel sheet. We can still scale up by using multiple sheets. However, doing so will cause Excel's data processing speed to drop, and looking up for any single rerecord of data will become a time-consuming task. On the other hand, an RDBMS can process millions of records of data without losing speed.
- Multiple users can access and work on a single database simultaneously over a network or on the Internet. Although spreadsheets are originally designed for a single user, an updated version of a spreadsheet such as Google Sheets can also provide multi-user functionality. For a database, multiple users can update or manipulate the data at the same time, but the RDBMS will carry out only a single process at any point in time. For a spreadsheet like Google Sheets, however, a multi-user session can be confusing. It can also cause data integrity loss because data can be deleted, overwritten, or changed simultaneously in unintended ways due to human errors. This error type will not happen in an RDBMS because the data input type is predefined during the database design process. Human error is eliminated because, in a database, each record in a table will not accept data of a type it is not assigned to keep. Therefore, a database can preserve data integrity better than any usual spreadsheet software.
- SQL and an RDBMS such as MySQL can help us with many data-related tasks; however, they cannot perform complex data analysis and visualization by themselves as we can do in Excel.

Ques9 – What is normalization ? Why it is considered so important in Database theory? and different types of normalization 1NF , 2NF, 3NF or BCNF

Normalization is basically to design a database schema such that duplicate and redundant data is avoided. If the same information is repeated in multiple places in the database, there is the risk that it is updated in one place but not the other, leading to data corruption. Though it is helpful while creating database design it is one of the most important aspect to ensure the data integrity , as if my data is not normalized, due to redundancy the volume of the data will increase which will directly effect my processing power , as well as my queries may give us wrong results . we also need to understand that we can never completely remove the redundancy from our database since it's a building block to make a database relational, we can only remove the unnecessary redundancy by the process of normalization.

1st normal form :

Each table cell should contain a single value.

Each record needs to be unique.

2nd Normal Form : Rule 1- it should be in 1NF

Rule 2- Single Column Primary Key that does not functionally dependant on any subset of caQndidate key relation.

3rd Normal Form:

Rule 1- table should be in 2NF

Rule 2- it Has no transitive functional dependencies

BCNF (Boyce Codd Normal Form):

A relation is in BCNF, if and only if, every determinant is a Form (BCNF) candidate key.

There are many technical jargons which we used to explain normalization in a proper way.

You can learn more about them here:
https://www.guru99.com/database-normalization.html

Question10 - is foreign key important to create joins? Yes or no? What will happen if you have not created a relation of primary – foreign key between two columns and trying to create joins out of that.

Foreign keys are a preferred way to create joins between two tables, but they are not compulsory to create joins. Foreign key maintains the referential integrity of the table. Referential integrity is defined as the degree to which data in two or more tables related through a foreign key relationship is complete. Referential integrity is often explained through parent/child table relationships. Within a relational database, all values that are present on a child table should also be present on its parent table (the child inherits values from the parent). If a child table has values that are not on its parent table, it does not have referential integrity with the parent table.

We can create joins without even foreign key – primary key relationship , but then the referential integrity won't be maintained.

Question11- What is OLAP? How is it different from OLTP?

OLAP, or Online Analytical Processing, databases store data in an aggregated form from multiple OLTP databases. This data is then stored within a data warehouse. But, instead of a transaction-level view, it gives a multidimensional view of the data. This means that if the organization wants to view the aggregated sales data, they can view them according to multiple categories and subcategories- location (region, country, state), time (year, month, day), the customer (gender, age), etc. This enables organizations to perform advanced analytics on their data, thereby giving a deeper understanding of their products. So, basically this data is stored to generate data driven insights for the organization.

OLTP, or Online Transactional Processing, systems handle a large number of transactions happening in real-time. But, what are the transactions?

Well, Transactions are processes that occur in their entirety and in isolation from one another. They either insert, update, or delete data in a database. On successful execution, the changes made by a transaction to a database persist in the database even in the event of a system failure.

The transactional data is stored in Relational Databases that ensure ACID properties for transactions. This data is written and queried at a very high pace to prevent any delay in processing OLTP governs transactions because they are the critical processes that we encounter in our everyday life. Online transactions, e-commerce orderings, online hotel bookings, ATM transactions, etc. are all managed by OLTP processes.

Question12- What are joins in SQL ? Why are they so important?

Join plays an important part in Structure Query Language.Joins are used when one needs to combine data from various tables. We normalize big tables to form smaller tables to avoid various anomalies. Now for example we have to develop an application where data from these small tables has to be displayed as one single entity. Here the joins come into picture where we can join various tables to display the data**.**

This is considered as one of the most important concepts in SQL, since while retrieving the data in the real world you will almost never fetch the data from a single table. You almost always need to join multiple tables in order to get the desired results.

Question 13 and 14) Different types of joins, and how they work.

Ans - A more intuitive explanation can be found here:
Joins - https://www.w3schools.com/sql/sql_join.asp

Inner Joins - https://www.w3schools.com/sql/sql_join_inner.asp

Left Join - https://www.w3schools.com/sql/sql_join_left.asp

Right join - https://www.w3schools.com/sql/sql_join_right.asp

Full Join - https://www.w3schools.com/sql/sql_join_full.asp

Self Join- https://www.w3schools.com/sql/sql_join_self.asp

Question15 – What is subquery? Why do we use them?

Ans - A subquery is a query that is nested inside another query. It is generally used in the where or having clause of an SQL statement. Subqueries basically let you specify the results of one query as an argument in another query. They are also known as Inner Query or the Nested Query.

Important : It is also very important to know when we should use subquery V/S joins.

You can learn more about here:

https://learnsql.com/blog/subquery-vs-join/

Another Important thread you should look into :

https://stackoverflow.com/questions/4799820/when-to-use-sql-sub-queries-versus-a-standard-join

Question16 – What are aggregate functions in SQL?

An aggregate function performs a calculation on a set of values, and returns a single value. For Example you want to calculate how many no of rows there are in your table you will use count() , which will return a single value i.e. the no of rows.

There are five aggregate function that we can use in SQL

1 Count
2 SUM
3 AVG
4 MIN
5 MAX

Question17 - What is Indexing? Name the different types of indexes in SQL and define them.

Indexing is the way SQL helps us perform fast searches and queries. It Basically is a data structure that improves the speed of operations in a table. They can be created using one or more columns . Though indexing a column improves search but increases the time for insert and update.

So, we index only columns with really frequent uses . otherwise, it may affect the overall user performance.

You can think about it as a library, new books come, some are updated or removed and searching books is an easy task , so search time decreases but the insertion and update time increases.

There are various types of indexes in SQL

1 Clustered Index
2 Non-Clustered Index
3 Column Store Index
4 Filtered Index

1 **Clustered Index** - Clustered Index store and sort rows of data in a view or table depending on their central values. There may be an instance of having just one clustered index in each table, as it can empower the client to store data in a solitary request. Clustered index store data in an arranged way, and in this way, at whatever point data is contained in the table in an arranged manner implies it is orchestrated with a clustered index.
2 **Non-Clustered Index**- A non-clustered index (or regular b-tree index) is an index where the order of the rows does not match the physical order of the actual data. It is instead ordered by the columns that make up the index. In a non-clustered index, the leaf pages of the index do not contain any actual data, but instead contain pointers to the actual data. These pointers would point to the clustered index data page where the actual data exists.
3 **Column Store Index**- The column-store index empowers putting away information inside little impressions, which helps in speeding up. The use of this index takes into account the client to get IO with multiple times higher inquiry execution when contrasted with conventional column arranged capacity. For examination, the Columnstore Index gives a significant degree to have a preferable exhibition over other records in SQL. Column store index esteems from a similar area have comparative qualities, which expands the general pace of information compressions.
4 **Filtered Index** - A filtered index is one of the types of indexes in SQL server that is made when a column has just a few applicable numbers for questions on the

subset of values. If, when a table comprises heterogeneous data rows, a separated list is made in SQL for at least one sort of data.

Question18- What are the advantages of creating a view in a SQL database?

1. Views allow you to create a stored way of looking at the information in your table.When you create a view, you create a way of querying the database table not only today, but also in the future.

2. You can come back to a view and recall it, producing updated results.

3. A simple view can be thought of as a subset of a table and can be used for retrieving data, as well as updating or deleting rows.

4. Rows updated or deleted in the view are updated or deleted in the table the view was created with.

5. Views allow the database administrator (DBA) to pull fields of interest from tables of interest and return a coherent data set useful to some specific user or application.

Question19- What is the order of execution in a query?

The order of query goes like this:-

FROM – Choose and join tables to get the raw data

WHERE – First filtering condition

GROUP BY – Aggregates the base data

HAVING – Apply condition on the base data

SELECT – Return the final data

ORDER BY – Sort the final data

LIMIT – Apply limit to the returned data

Ques20 - What is the difference between NVL and NVL2?

In SQL, NVL() converts a null value to an actual value. Data types that can be used are date, character and number. Data type must match with each other i.e. expr1 and expr2 must of same data type.

NVL (expr1, expr2)

expr1 is the source value or expression that may contain a null.

expr2 is the target value for converting the null.

NVL2(expr1, expr2, expr3) : The NVL2 function examines the first expression. If the first expression is not null, then the NVL2 function returns the second expression. If the first expression is null, then the third expression is returned i.e. If expr1 is not null, NVL2 returns expr2. If expr1 is null, NVL2 returns expr3. The argument expr1 can have any data type.

NVL2 (expr1, expr2, expr3)

expr1 is the source value or expression that may contain null

expr2 is the value returned if expr1 is not null

expr3 is the value returned if expr2 is null

Question21- What is the use of the FETCH command?

The FETCH argument is used to return a set of rows. FETCH can't be used itself, it is used in conjunction with OFFSET.

```
SELECT column_name(s)

FROM table_name

ORDER BY column_name

OFFSET rows_to_skip

FETCH NEXT number_of_rows ROWS ONLY;
```

Question22 - What is the use of offset?

Offset is used to specify from which row we want the data to retrieve. To be precise, specify which row to start retrieving from. Offset is used along with the LIMIT. Here, LIMIT is nothing but to restrict the number of rows from the output. OFFSET can only be used with an ORDER BY clause. It cannot be used on its own. OFFSET value must be greater than or equal to zero. It cannot be negative, else return error.

For example – Let's say you want to find the 2nd highest salary from a mysql db, one way to use offset could be-

SELECT * FROM employees

ORDER BY ctc DESC

LIMIT 1 OFFSET 1;

Question23) What is the NTILE() function?

Ans - NTILE() function distributes the rows in an ordered partition into a specific number of groups. These groups are numbered. For example, NTILE(5) will divide a result set of 10 records into 5 groups with 2 records per group.

If the number of records is not divided equally in the given group, the function will set more records to the starting groups and less to the following groups.

SELECT emp.*,

NTILE(3) over (order by salary DESC) as GeneratedRank

from Employee emp;

This will divide the complete data set in 3 groups from top based upon the salary in descending order.

Ques-24) What is the RANK() function?

The RANK() function assigns a rank to each row within the partition of a result set. The rank of a row is specified by one plus the number of ranks that come before it.

The basic Syntax for this is as follows-

RANK() OVER (

    PARTITION BY <expression>[{,<expression>...}]

    ORDER BY <expression> [ASC|DESC], [{,<expression>...}]

)

Ques- 25) What is DENSE_RANK()?

This gives the rank of each row within a result set partition, with no gaps in the ranking values. Basically there is no gap, so if the top 2 employees have the same salary then they will get the same rank i.e. 1 , much like the RANK() function. But, the third person will get a rank of 2 in DENSE_RANK as there is no gap in ranking where as the third person will get a rank of 3 when we use RANK() function. Syntax below:-

SELECT emp.*,

DENSE_RANK() OVER (order by salary DESC) DenseRank

from Employee emp;

Ques-26) What is the difference between View and Store Procedure?

A view is a “virtual” table whose data is the result of a stored query. It is derived each time you query against the view. It allows selection of only particular columns or only particular rows of data so is useful in minimizing the storage required for result sets. It may also be used to enforce security rules.

A stored procedure is a set of SQL commands that perform specific functions. It is compiled once and may be re-executed. It may be designed to act on input parameter values.

Alternatively,

View

1 Does not accept parameters

2 Can be used in FROM clause. Can be used as a building block in

3 larger query

4 Contains only Select query

5 Cannot perform modification to any table

Store Procedure

1 Accept Parameters

2 Cannot be used in FROM clause. Hence, cannot be used a building

3 block in larger query

4 Can contains several statements, IF-ELSE, Loop etc
5 Can perform modification to one or several tables

Ques-27) What is partitioning in SQL?
Partitioning is the database 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 less data to scan. The main goal of partitioning is to aid in maintenance of large tables and to reduce the overall response time to read and load data for particular SQL operations.
Alternatively,
SQL Server supports table and index partitioning. Partitioning is a way to divide a large table into smaller, more manageable parts without having to create separate tables for each part. Data in a partitioned table is physically stored in groups of rows called partitions and each partition can be accessed and maintained separately. Partitioning is not visible to end users, a partitioned table behaves like one logical table when queried.

Ques-28) What is a Candidate Key?

Candidate key is a single key or a group of multiple keys that uniquely identify rows in a table.

A Candidate key is a subset of Super keys and is devoid of any unnecessary attributes that are not important for uniquely identifying tuples.

The value for the Candidate key is unique and non-null for all tuples. And every table has to have at least one Candidate key. But there can be more than one Candidate Key too.

Ques-29) What is NULLIF?

The MySQL **NULLIF()** function is used for the comparison of two expressions. The NULLIF() function returns NULL if both the expressions are equal, else it returns the first expression. The NULLIF() function accepts the expressions as parameters and returns NULL if both of them are equal.

Ques-30) What is ROW_NUMBER()?

The ROW_NUMBER() is a window function that assigns a sequential integer number to each row in the query's result set.

The following illustrates the syntax of the ROW_NUMBER() function:

```
ROW_NUMBER() OVER (

   [PARTITION BY expr1, expr2,... exprn]

   ORDER BY expr1 [ASC | DESC], expr2,...

)
```

In this syntax,

- First, the PARTITION BY clause divides the result set returned from the FROM clause into partitions. The PARTITION BY clause is optional. If you omit it, the whole result set is treated as a single partition.
- Then, the ORDER BY clause sorts the rows in each partition. Because the ROW_NUMBER() is an order sensitive function, the ORDER BY clause is required.
- Finally, each row in each partition is assigned a sequential integer number called a row number. The row number is reset whenever the partition boundary is crossed.

Ques-31) What are the important conditions for joining two tables on a key?

Ans - Ideally there should be one common column(at least) between the tables like Roll No. in Student and Class_Topper Table (Assuming table names), and the datatype of those columns should also be the same.

Ques-32) What is lag() function? How is it different from the lead() function?

Lag() Provides access to a row at a given physical offset that comes before the current row. We Use this function in a SELECT statement to compare values in the current row with values in a previous row as specified by offset. Default offset is 1 if not specified. If Partition By clause is specified then it returns the offset Value in each partition after ordering the partition by Order By Clause.

Whereas,

Lead() Provides access to a row at a given physical offset that comes after the current row. Use this function in a SELECT statement to compare values in the current row with values in a subsequent row as specified by offset. Default offset is 1 if not specified. If Partition By clause is specified then it returns the offset Value in each partition after ordering the partition by Order By Clause.

Ques-33) What is the use of INSTR and SUBSTR function?

The INSTR function returns the position of the first occurrence of a substring in a string. If the substring is not found in the str, the INSTR function returns zero (0).

The SUBSTR() function extracts a substring from a string (starting at any position). If the specified length value is longer than the input string, the result is the full input string.

Ques-34) How is data modeling different from database design?

Data Modeling: It can be considered as the first step towards the design of a database. Data modeling creates a conceptual model based on the relationship between various data models. The process involves moving from the conceptual stage to the logical model to the physical schema. It involves the systematic method of applying the data modeling techniques.

Database Design: This is the process of designing the database. The database design creates an output which is a detailed data model of the database. Strictly speaking database design includes the detailed logical model of a database but it can also include physical design choices and storage

parameters.

Theory +Logic

(We strictly suggest you go through the basic theoretical part first, because if you are clear with that basic concept, many questions here will just need some logic over those same theoretical concepts.)

Question1 - When will ROW_NUMBER and RANK give different results?

Both the functions ROW_NUMBER and RANK return an integer value that are incremental in nature. ROW_NUMBER as the name suggests assigns an incremental number to each subsequent row whereas RANK functions rank the records by values imposed by the ORDER BY clause.

However, both these functions differ when there are records that are tied in values of the columns stated in the ORDER BY clause or you can say if not all values are unique that column.

When records are tied, ROW_NUMBER will simply return the corresponding row number whereas RANK will return the values of the previous row in case of a tie and also skip the rank.

Question -2 - Is it possible for LEFT JOIN and FULL OUTER JOIN to produce the same results? Why or why not?

Yes, It's possible that Left outer join and full outer join produce the same results . Consider the two cases to understand this better which are as follows –

1 There are no rows in table A and B that do not obey the match condition.
2 There are no rows in B that do not have a match in A.

Question3 - Why would I use DENSE_RANK instead of RANK? What about RANK instead of DENSE_RANK?

DENSE_RANK and RANK are functions used to rank records. The only key difference between them is that while RANK skips numbers/rank in case of ties whereas DENSE_RANK doesn't skip numbers/rank and assigns consecutive numbers.

In a real world scenario, we can use both to identify the nth highest salary or certain metric, however, the business case for how to treat in case of ties would differentiate which functions to use over the other.

So, In general we can say,

1 Use RANK() when it is important only to identify the 3rd highest regardless if there are ties or not.
2 Use DENSE_RANK() when it is important to identify the 3rd highest taking into account that there could be some salesperson who have same rank.

Question 4- What happens if I GROUP BY a column that is not in the SELECT statement? Why does this happen?

The correct answer to this question is subjective to the type of rdbms you are working with .

For example – if you are working with oracle or MS-SQL server, When a GROUP BY column that is not in the SELECT statement is used, it will throw an error.

Because By the order of operations in these rdbms, SELECT or selecting columns run first before SQL groups them. If a column that is not in select statement is used, GROUP BY cannot locate the column and hence it will throw an error.

However it won’t throw any error if you are trying to perform the same task in Mysql.

Question5 - LAG and LEAD are especially useful in what type of scenarios?

LAG and LEAD are window functions that is always followed by an OVER clause. LAG function returns the preceding value from the current row whereas LEAD function returns the succeeding value from the current row.

LAG and LEAD are window functions that is always followed by an OVER clause. LAG function returns the preceding value from the current row whereas LEAD function returns the succeeding value from the current row.

In order to understand their detailed use cases, check out this excellent article on them.

Question 6- For dealing with NULL values, when would you choose to use IFNULL vs. CASE WHEN?

There are three consideration we should ideally look out for when deciding to choose to use IFNULL vs CASE WHEN:

1. IFNULL supports only one condition whereas CASE WHEN supports multiple conditions. You could have multiple nested IFNULL however, this leads me to my next point.
2. For complex conditions, CASE WHEN eases readability than IFNULL functions.
3. Another thing on which we should keep an eye on is their execution time.

Question7 - Do temp tables make your code cleaner and faster, one of the two, or none? Why?

Temp tables are storage tables that exists only temporarily in database. When the connection that created the temp tables is closed, it ceases to exist. Depending on the type of temp tables for instance with global temp tables, it is possible for multiple connections/sessions to call the temp tables.

In my opinion, temp tables do make the code cleaner in respect to making it easier to follow the logic and understand it. Some of the articles do suggest that using temp tables can make the processing faster as we can use temp tables to process and store complex sql queries which then can be references by other sql statements.

Question8 - When is a subquery a bad idea? A good idea?

Ans - Subqueries are a bad idea

when it is used as a correlated subqueries and there are a large number of rows returned by the parent query. In these instances, correlated subqueries execute once for each row returned by the parent query. For example, when the parent query has 1000 rows, the correlated query also runs each time for the 1000 rows. This can be taxing performance wise and it is best to use other methods that could save the performance.

Subqueries are a good idea to use

when there is no need to run dependent queries. Unlike correlated or dependent queries, independent queries will only be evaluated once.

Question-9) OUTER (e.g. LEFT) and INNER JOIN when/where to use them.

The major difference between inner and outer joins is that inner joins result in the intersection of two tables, whereas outer joins result in the union of two tables.

So ,basically in order to perform a particular task if you want only the rows which are common in both the tables, we should use the inner join.

Whereas, outer joins are useful if you want to get all the values from one table but only the rows that match that table from the other table.

For example, the left join returns all the values from the left table and only the common rows from the other table.

Question10 - UNION V/S UNION ALL, which one of them is more efficient?

The UNION operator removes or eliminates duplicate rows, whereas the UNION ALL operator does not.

So there is a specific amount of time or the execution power is required to remove duplicates while running the UNION command, hence union all execute faster than that of the union.

Question11 - When are COUNT(*) and COUNT DISTINCT equal?

Count would show a result of all records while count distinct will result in showing only distinct count.

So, basically whenever all the records available in my table are unique , count and count distinct will return the same output.

Question12 - Can left and inner join return the same results?

Yes, its possible for left and INNER joins to return the same results when

- When number of entries in joining column are same in both the table
- No. of entries in the joining column in the right table are more than that of the left table.

Question13 – What is the difference between RANK , DENSE RANK and Row number?

The row_number gives continuous numbers as an output, while rank and dense_rank give the same rank for duplicate values, but the next number in rank is as per continuous order so you will see a jump if you are working with rank but dense_rank doesn't have any gap in rankings.

Question14 - Delete V/S Drop V/S Truncate

Delete is basically a DML(Data Manipulation Language) Command, which is generally used with WHERE clause , in order to delete the records from the table according to some condition, if we don't specify where clause along with Delete , all the records of my table will get deleted.

DROP is basically a DDL(Data definition language) command , which is used to delete a database or a table from the database.

Truncate basically gives the output similar to using DELETE without WHERE clause , but it's a DDL command because of the way it works internally , it first delete the whole tale and then put back the structure of the table due to which we get the empty table with the predefined structure.

Question15 - What are the best scenarios to use a self join?

A self join allows you to join a table to itself. It helps query hierarchical data or compare rows within the same table or if you want to do some comparison within the table.

Suppose that there exists a database table called Chats that holds all of the chat messages that have been sent or received by an online clothing store business.

It would be extremely beneficial for the clothing store owner to know how long it usually takes her to respond to messages from her customers.

But, the messages from her customers and messages to her customers are in the same data source. So, we can use a self join to query the data and provide this analysis to the store owner. We will need one copy of the Chats table to get the initial message from the customer and one copy of the Chats table to get the response from the owner.

Then, we can do some date math on the dates associated with those events to figure out how long the store owner is taking to respond.

Question16 – How can you use SQL to dedupe rows?

Dedupe(process of deduplication or removing the duplicates)

By using CTE's and Ranking window function, we can dedupe a database in SQL.

You can learn more about this here .

Question17 - In what scenarios is the LAG function useful?

LAG function is especially useful when you want to compare the current value with the previous value for some computation.

For example – they can be useful in Time series forecasting to make time series model stationary in order to implement ARIMA models.

Question18 - Why use the WITH statement? What does it help you do?

The with Statement allows us to alias a particular query block which we can reuse multiple times just by referring to the alias, which acts as a temporary virtual table for us. It also helps us in writing complex blocks of query once and reuse multiple times, avoid repetition , better readability and understandability also that complex query is executed only once and that result set is reused again, so it gives better performance as well.

Question19 - Can you use COUNT DISTINCT on two columns? How?

Yes, we can use COUNT DISTINCT on two columns

In order to do so , we can either use SELECT COUNT DISTINCT(col1, col2)

Alternatively we can also use

Concatenating the columns and select the count (count(distinct(col1||col2))

Question20 - Use a window function to calculate a running total.

We can use the window function sum with rows unbounded preceding in order to calculate the running totals. A general syntax that can be followed in such case is as follows –

window_function (column)

OVER ([PARTITION BY partition_list] [ORDER BY order_list])

Question21 - Why would you use GROUP BY ROLLUP?

GROUP BY ROLLUP is an extension of the GROUP BY clause that produces subtotal rows (in addition to the grouped rows). Sub-total rows are rows that further aggregate whose values are derived by computing the same aggregate functions that were used to produce the grouped rows.

You can think of rollup as generating multiple result sets, each of which (after the first) is the aggregate of the previous result set.

Question22 - If an error is thrown using SQL, where do you start looking?

In order to solve an error we usually get an idea of the error from the GUI itself as it throws a select failed error followed by a generic syntax error. Check what the error says and go from there. Apart from that try to see if there is an specific error code mention wrt that error to debug.

Question23 – In what scenarios the lead() function will be useful?

It’s useful to look for an event /row that happened after the current event. This can be generalized to pull data from n+1 rows (where n being the current row).

For example the next highest revenue generating business unit.

Question24 - When does a full outer join actually make sense to use?

There can be various use case, in which we can use full outer join.

One of them can be for the comparison of the records basically to compare what is in (Table A and not in Table B) and vice versa or may be to find the mutually exclusive records.

Question25 - What is the difference between Having and where?

Having is basically used to filter the records at the aggregation level whereas the where clause is used to filter the records at the row level or at the individual level.

As per the order of the execution in SQL, where is executed first then group by so basically after the execution of group by, if there is any kind of filtering needs to be done, then we have to use the where clause for the same.

Question26 - If you want to allow ties in ranking, RANK or DENSE RANK?

OR RANK , DENSE_RANK or ROW_NUMBER , when to use which?

Both of them allow ties in ranking, the only difference is DENSE_RANK() results will be continuous in nature(1,1,1,2,,3,3,) whereas in RANK() the result will not be continuous in nature(1,1,1,4,5,5,7)

Whereas,

If we are trying to use ROW_NUMBER, it won’t allow any ties in ranking even if the column values are the same.

Question27- Can UNION and UNION ALL return the same results?

Yes, they both can give the same results.

So, basically UNION ALL returns the duplicates as well whereas UNION does not, so if all the records are already unique in the table, then UNION and UNION ALL will return the same results.

Question28 – When is it a bad idea to use a subquery?

If you have a dependent subquery you might run into performance problems because a dependent subquery typically needs to be run once for each row in the outer query. So if your outer query has 1000 rows, the subquery will be run 1000 times. Though, sometimes the performance of a subquery also depends upon the type of RDBMS you are working with.

Question29 - Inner joining tables A and B returns 50 rows. Left join B to A returns 125 rows. What's going on here?

Since, Inner joining these two tables gives 50 records that means there are 50 records which are common in both the tables.

Whereas left join B to A gives 125 rows that means there are total 125 rows in B and out of which 50 are common in both the tables and 75 rows are there only in table B so its mutually exclusive .

Question30 - I wanted to see the distribution of a variable in SQL, how can i do that?

The result of a group by query is actually the distribution of my resultset according to some category, so whenever we want to see the distribution we can actually use the group by clause with the count function.

Question31 - How can you remove duplicate rows without using distinct keywords?

There can be mainly two ways by which we can perform this task without using distinct keyword:

1 Using row_number and CTE's – this is one of the many ways possible, so basically we create a CTE and then use row_number partition to delete the duplicate records as filters.
2 Group by – another way by which same task can be done is to group the column by the most unique entity we have in the database (for example employee id in the employees database)

Question32 - How would you create an empty table using an existing table?

```
CREATE TABLE new_table AS (SELECT *

    FROM old_table WHERE 1=2);
```

In the inner query , since 1 is never equal to 2 , hence no record will be transferred in the new table but the structure will get copied. And we will have an empty table with the same structure as the old table.

Question33 - What is the use of ON DELETE CASCADE?

ON DELETE CASCADE is a clause generally used with foreign keys to maintain the integrity between child and the parent table.

For example – let's just say you are working with a company who gives its monthly subscription to its users. So there are two tables basically one for the customer_details and another one for the subscription details. Now let's say one of your customer has recently opt out from the subscription plan. So you may want to remove his name from the customer_details table, ON DELETE CASCADE will let you do that automatically.

Question34 - What is the difference between COUNT(*) and COUNT(ColName)?

COUNT(*) will return the total no of rows including the rows which have null's in them. Whereas if you are using count(colname) it will only return the count of non –null values.

So, basically if my dataset does not contain any null values in any of its columns , then both will give the same results.

Question35 – How does Partition BY work?

Or question -36 What is the difference between partition by and group by?

The PARTITION BY clause is a subclause of the OVER clause. The PARTITION BY clause divides a query's result set into different partitions and then further window function is operated on each partition separately and recalculate for each partition.

The general syntax of the partition by is as follows –

```
window_function ( expression ) OVER (
    PARTITION BY expression1, expression2, ...
    order_clause
    frame_clause
)
```

In order to understand the difference between partition by and group by check out this.

Question37 - Can we have another column in a table other than a primary key column which will act as a primary key?

According to the rules of DBMS, we can only have one primary key in the table, but we can always declare a column as NOT NULL + UNIQUE, and it will behave in the same manner as our primary key does.

PRACTICAL QUESTIONS

(This section contains practical query related questions from beginner to intermediate level.)

Q1. Write a query to get employee name starting and ending with vowels

```
SELECT EmpID,EmpName

FROM Employee

where EmpName like ‘[aeiou]%[aeiou]’
```

Q2. How to fetch only common records between two tables?

```
SELECT * FROM table
INTERSECT
SELECT * FROM table1
```

Q3. Consider a table T1 which contains ‘Column A’ in it. Column A is declared as a primary key. Which of the following is true for column “A” in T1?

Column A can contain values [1,2,3,4,1]
Column A cannot contain values [1,2,3,4,1]
Column A can contain values [1,2,3,4,NULL]
Column A cannot contain values [1,2,3,4,NULL]

Ans – B, D

Q4. We have two columns(Revenue and Cost Price) in a table like below, get me the Profit column given that direct adding the column values will not work.

Revenue	**Cost Price**	**Profit**
100	12	88
192	NULL	192
187	13	174
NULL	50	-50

```
Select (coalesce(Revenue,0) + coalesce(CostPrice,0))
as Profit From Table
```

Alternatively, we can also use ifnull() to solve the same query.

Q5. You Have been given "Employee" table with the structure as follows-

Employee Name	Department	Salary

1 Highest Salary from the table

SELECT MAX(Salary)

FROM Employee;

2 Second Highest from the table

SELECT Salary FROM TABLE

ORDER BY Salary DESC

LIMIT 1 OFFSET 1;

Alternatively , another way in which it can handle duplicacy as well

SELECT MAX(salary) FROM Employee

WHERE salary < (SELECT MAX(salary) FROM employee);

3 Department wise Highest Salary

SELECT Department, MAX(Salary) FROM Employee

GROUP BY Department;

4 Department wise second highest salary

select t.dept, max(t.salary) as maxs

from employee t

```
where t.salary < (select max(salary)
                  from employee t2
                  where t2.dept = t.dept
                 )
group by t.dept;
```

5 Get all the details of the person with 3rd highest salary?

```
SELECT * FROM employee
 WHERE salary = (SELECT Salary FROM employee
ORDER BY SALARY DESC LIMIT 1 OFFSET 2);
```

6 Get the name of all the employees whose name start with any letter between a - f?

```
SELECT name FROM employees
WHERE name LIKE ‘[a-k]%’;
```

Ques-6) select case when null=null then ‘Rahul’ else ‘Akash’ end from dual

If you are comparing with NULL , it will always give the false as result, hence the else statement will get executed , and the output will be ‘Akash’.

Ques-7) Consider the employee table with the following schema -

Emp_number	Employee Name	Job	Manager ID	Salary	Hire_date	Commision	Department_number

5.1 List the employees who joined before 2015

```
SELECT * FROM employee
WHERE Hire_Date < ’01-Jan-2015’;
```

5.2 List the employees whose annual salary is between 300000 and 450000

```
SELECT * FROM employee

WHERE Salary*12 BETWEEN 300000 AND 450000;
```

5.3 List the employees who joined in January

```
SELECT * FROM employee

WHERE to_char('hire_date', 'MON') IN ('Jan');
```

5.4 List the employees who are senior to their own Manager

```
SELECT * FROM employee w, employee m

 WHERE w.manager_id = m.Employee_no

and w.Hire_Date < m.Hire_Date;
```

5.5 Get the details of the senior most employee.

```
SELECT * FROM employee

WHERE hire_date = (SELECT hire_date FROM employees

ORDER BY hire_date LIMIT 1);
```

6 Find the total salary given to Manager

```
SELECT SUM(Salary) FROM employees

WHERE Employeeid IN (SELECT DISTINCT manager_id FROM employees);
```

7 Which department have the highest average salary

```
 SELECT dept_no,  AVG(Salary) FROM employees
```

```
            GROUP BY dept_no  ORDER BY AVG(salary)
LIMIT 1;
```

8 How to find Third highest salary in Employee table using self-join?

```
SELECT DISTINCT salary
  FROM employee a
WHERE 3 >= (SELECT COUNT(DISTINCT salary)
              FROM employee b
            WHERE a.salary <= b.salary)
ORDER BY a.salary DESC;
```

Ques8- Write a Query to find the count of duplicate rows from a 'student' table.

```
SELECT COUNT(roll_no) FROM Student
GROUP BY roll_no
HAVING COUNT(roll_no) > 1;
```

Ques-9 - How to calculate number of rows in table without using count function?

```
SELECT table_name, num_rows
FROM user_tables
WHERE table_name='Employee';
```

Ques-10. Consider the table 'Payment' with the following schema :

Payment_id	Customer_id	Staff_id	rental_id	amount	paid_at
Integer	smallint	smallint	integer	numeric	Timestamp

1.Write a query to return the total movie rental revenue for each month.

```
SELECT EXTRACT(YEAR FROM payment_ts) AS year
, EXTRACT(MONTH FROM payment_ts) AS mon,
 SUM(amount) as rev FROM payment GROUP BY year,
 mon ORDER BY year, mon;
```

2. write a query which will return the daily revenue in june 2021.

```
SELECT DATE(payment_ts) AS dt,
 SUM(amount) FROM payment
 WHERE EXTRACT(MONTH FROM payment_ts) = '6' AND
 EXTRACT(YEAR FROM payment_ts) = '2021'
GROUP BY dt ORDER BY dt;
```

3. Average customer spend by month

```
SELECT EXTRACT(YEAR FROM payment_ts) AS year,
 EXTRACT(MONTH FROM payment_ts) AS mon,
 SUM(amount)/COUNT(DISTINCT customer_id) AS avg_spend
 FROM payment GROUP BY year,
 mon ORDER BY year, mon;
```

4.Write a query to return the month with lowest total rental revenue

```
SELECT EXTRACT(MONTH FROM payment_ts) AS mon,
SUM(amount) AS lowest_revenue

FROM payment GROUP BY mon

ORDER BY MON DESC

LIMIT 1;
```

5. Write a query to count the number of customers who spend more than (>) 40k by month

```
SELECT year, mon, COUNT(DISTINCT customer_id)

 FROM ( SELECT EXTRACT(YEAR FROM payment_ts) AS year,

 EXTRACT(MONTH FROM payment_ts) AS mon,

  customer_id, SUM(amount) amt FROM payment GROUP BY year, mon, customer_id ) X

WHERE amt > 40,000

 GROUP BY 1,2;
```

6 Write a query to return the minimum and maximum customer total spend in June 2020.

```
WITH cust_tot_amt AS (

SELECT

customer_id,

SUM(amount) AS tot_amt

FROM payment

WHERE DATE(payment_ts) >= '2020-06-01'

AND DATE(payment_ts) <= '2020-06-30'
GROUP BY customer_id

)

SELECT

MIN(tot_amt) AS min_spend,

MAX(tot_amt) AS max_spend

FROM cust_tot_amt;
```

CLOUDYML