MySQL Interview

Questions & Answers E-BOOK

By Gurunatha Dogi

www.questpond.com

Contents

Q1: Explain Normalization & Denormalization?

Normalization is a database design technique to remove redundant data and eliminates undesirable characteristics like Insertion, Update and Deletion Anomalies. We have types of Normal forms like 1st normal Form, 2nd Normal Form and 3rd normal Form which are used to eliminate or reduce redundancy in database tables. Normalization rules says divide larger tables into smaller tables and links them using joins.

Denormalization is a database design technique which focuses on improving search performance. Here we go opposite of normalization that is here we merges two tables in-order to reduce no of joins which helps in speeding up the performance. In simple words fetching queries can be simpler in denormalization because we need to look at fewer tables.

For example, on normalization and denormalization please watch video.

Q2: Explain 1st, 2nd and 3rd Normal Form in MYSQL Database Normalization?

Normalization is a database design technique to remove redundant data and eliminates undesirable characteristics like Insertion, Update and Deletion Anomalies. We have types of Normal forms like 1st normal Form, 2nd Normal Form and 3rd normal Form which are used to eliminate or reduce redundancy in database tables. Normalization rules says divide larger tables into smaller tables and links them using joins.

1st Normal Form (1 NF): If a table is to be in 1 NF, then every table must have Primary Key, Column must have Atomic Value and there should not be repeating group problem.

2nd Normal Form (2 NF): 1 NF + Remove Partial Dependency.

3rd Normal Form (3 VF): 1 NF + 2 NF + Remove Transient Dependency.

For example, kindly watch video of question 2.

Q3: Primary key vs foreign key?

Primary Key	Foreign Key
Primary key focuses on uniqueness in the table	A foreign key is a column in the table that is the primary key in another table.
Primary Key cannot be null and it should be integer	Foreign Key can be null and it should be integer
We can have one Primary Key	We can multiple foreign keys in a table
To identify any record primary key is must and every table must have primary key	A foreign key is a field in the table that is the primary key in another table and it is not compulsory to have in every table.

Q4: Primary key vs Candidate key vs Unique key?

Primary Key	Candidate Key	Unique Key
Primary key focuses on uniqueness in the table	Candidate key focuses on uniqueness in the table and offers single key or a group of multiple keys that uniquely identify rows in a table	Unique key focuses on uniqueness in the table and offers single key or a group of multiple keys that uniquely identify rows in a table
Primary Key cannot be null and it should be integer	Candidate keys cannot be null and it can be both integer and string	Unique keys can be null and it can be both integer and string
We can have one Primary Key in a table	We can multiple candidate keys in a table	We can multiple unique keys in a table
To identify any record primary key is must and every table must have primary key	To identity any record we can use any one candidate key	To identity any record we can use any one unique key

Q5: What is a trigger? How you can create a trigger in MySQL?

A trigger is a special type of MySQL object or a type of stored procedure logic which gets triggers automatically when an event occurs like Insert / Update or Delete occurs on a table. This is very similar like a JavaScript event like onclick, onmousehover, on keypress and so on. so, you can write your logic for before insert or after insert / update / delete and so on.

Benefit of having a trigger is that you can track or you can log some information when insert / update / delete occurs to a table.

```
Drop Trigger If Exists tr_ins_location;
Create Trigger tr_ins_location
Before INSERT on `location`
For Each Row
SET NEW.datetime = Now();
```

Q6: Explain Transactions and How to Implement it in MYSQL?

Transaction helps us to execute group of statements like Select, Insert, Update and Delete as one logical unit that can be committed on successful and rolled back if any error occurs.

```
START TRANSACTION;

INSERT INTO `product` (`productname`,`productcost`) VALUES ('Headphones',350);
INSERT INTO `product` (`productname`,`productcost`) VALUES ('Bracelet',150);
INSERT INTO `product` (`productname`,`productcost`) VALUES ('Compass Box',50);

ROLLBACK;
```

Q7: What is an index? How can an index be declared in MySQL?

An Index in MySQL helps to improve speed of operations in a table.

So, for example consider any reading book or school text book. every book has an Index page with a page number so that reader can quickly can navigate to that particular chapter of page with scanning a whole book. Similarly

MySQL uses indexes to quickly find rows with specific column values. Without an index, MySQL must scan the whole table to locate the relevant rows for a given query.

Kindly note for Primary keys no need to create index.

For Creating Index

```
Create Index `ins` ON `product`(`productname`);
```

For Dropping Index

```
DROP INDEX `ins` ON `product`
```

Q8: What is the view? How can you create and drop view in MySQL?

The database views are the virtual tables that are generated by the query output. The sample example is shown below.

```
CREATE VIEW `propertylist` AS SELECT P.listingid, P.propertyname, L.location FROM `propertylisting` AS P INNER JOIN `location` AS L ON P.locationid = L.id;
```

As you see we have created view with view name "**propertylist**". Now to fetch records from view just write this below query.

```
SELECT * FROM `propertylist`
```

Q9: Explain Inner Join?

INNER JOIN helps us to make a join between two tables here output we will get only matching records from both tables. For example, we have OrderTable, ProductTable and another CustomerTable as shown below.

ProductTable

ProductID	ProductName	Cost
201	Bag	500
202	Jacket	800
203	Suit	1500

OrderTable

OrderID	ProductID	CustomerID	Quantity	TotalCost
101	201	1001	1	100
102	202	1005	2	1600
103	201	1010	1	500
104	202	1001	3	2400
105	201	1002	4	2000
106	203	1009	2	3000
107	202	1002	5	4000
108	203	1002	6	9000

CustomerTable

CustomerID	CustomerName	CustomerEmailID	CustomerSSNO
1001	Raju	abc@xyz.com	RJ1001
1002	Shiv	pqr@ybl.com	SH1002
1003	Khadak	cqrs@ppp.com	KH1003
1004	Sumesh	msn@abc.com	SM1004
1005	Rohit	zap@hgt.com	RH1005

If we apply INNER JOIN for all tables then:

```
Select Order.CustomerID, Customer.CustomerName, Order.ProductID,
Product.ProductName, Order.Quantity,Order.TotalCost, Order.OrderID From Order
INNER JOIN Customer ON Order.CustomerID = Customer.CustomerID
INNER JOIN Product ON Order.ProductID = Product.ProductID
```

INNERJOIN OUTPUT

CustomerID	CustomerName	ProductID	ProductName	Quantity	TotalCost	OrderID
1001	Raju	201	Bag	1	100	101
1005	Rohit	202	Jacket	2	1600	102
1001	Raju	202	Jacket	3	2400	104
1002	Shiv	201	Bag	4	2000	105
1002	Shiv	202	Jacket	5	4000	107
1002	Shiv	203	Suit	6	9000	108

Q10: Explain Left Join?

LEFT JOIN helps us to make a join between two tables here output we will get all records from left table and only matching records from right table. Let's take same example as we considered for INNER JOIN, we have ProductTable, OrderTable and another CustomerTable and if we apply left join in above tables then output will be.

```
Select Order.CustomerID, Customer.CustomerName, Order.ProductID,
Product.ProductName, Order.Quantity,Order.TotalCost, Order.OrderID From Order
LEFT JOIN Customer ON Order.CustomerID = Customer.CustomerID
LEFT JOIN Product ON Order.ProductID = Product.ProductID
```

Output

LEFTJOIN

CustomerID	CustomerName	ProductID	ProductName	Quantity	TotalCost	OrderID
1001	Raju	201	Bag	1	100	101
1005	Rohit	202	Jacket	2	1600	102
1010	Null	201	Jacket	1	500	103
1001	Shiv	202	Bag	3	2400	104
1002	Shiv	201	Jacket	4	2000	105
1009	Null	203	Suit	2	3000	106
1002	Shiv	202	Jacket	5	4000	107
1002	Shiv	203	Suit	6	9000	108

Q11: Explain Right Join?

RIGHT JOIN helps us to make a join between two tables here output we will get all records from right table and only matching records from left table. Let's take same example as we considered for INNER JOIN, we have ProductTable, OrderTable and another CustomerTable and if we apply right join in above tables then output will be.

```
Select Order.CustomerID, Customer.CustomerName, Order.ProductID,
Product.ProductName, Order.Quantity,Order.TotalCost, Order.OrderID From Order
RIGHT JOIN Customer ON Order.CustomerID = Customer.CustomerID
RIGHT JOIN Product ON Order.ProductID = Product.ProductID
```

Output

RIGHTJOIN

CustomerID	CustomerName	ProductID	ProductName	Quantity	TotalCost	OrderID
1001	Raju	201	Bag	1	100	101
1005	Rohit	202	Jacket	2	1600	102
1001	Shiv	202	Bag	3	2400	104
1002	Shiv	201	Jacket	4	2000	105
1002	Shiv	202	Jacket	5	4000	107
1002	Shiv	203	Suit	6	9000	108
1003	Sumesh	Null	Null	Null	Null	Null
1004	Khadak	Null	Null	Null	Null	Null

Q12: Explain Cross Join?

MySQL cross join produces result set from the no of rows in the first table multiplied no of rows from second table and this type of result set is also called as Cartesian product.

Here we can write Cross join with ON statement or without ON statement.

CourseTable

courseid	coursename
1	Science
2	Physics
3	Mathematics
4	Biology

StudentTable

studentid	studentname
1	Hari
2	Ram
3	Sumesh Joshi

```
SELECT * from `student` CROSS JOIN `course`;
```

CROSSJOIN OUTPUT

studentid	studentname	courseid	coursename
1	Hari	1	Science
2	Ram	1	Science
3	Sumesh Joshi	1	Science
1	Hari	2	Physics
2	Ram	2	Physics
3	Sumesh Joshi	2	Physics
1	Hari	3	Mathematics
2	Ram	3	Mathematics
3	Sumesh Joshi	3	Mathematics
1	Hari	4	Biology
2	Ram	4	Biology
3	Sumesh Joshi	4	Biology

Q13: MySQL Union vs Union All?

Union and Union All they both help us to combine two select statements and produce one result set but condition is that both tables must have same no of columns with same datatype then only they can product proper result set.

Exact difference between Union and Union All is that

Union does not display duplicate values. It only displays distinct values where else Union All displays all values including duplicates.

This is the only difference between union and union all.

Product1 Table

productid	productname	productcost
1	Computer	250
2	Television	350
3	Mobile	400
4	Laptop	500

Product2 Table

productid	productname	productcost

1	Smart Watch	100
2	Oven	150
3	Mobile	400

```
SELECT * FROM `product` UNION SELECT * FROM `product2`;
```

UNION Output

productid	productname	productcost
1	Computer	250
2	Television	350
3	Mobile	400
4	Laptop	500
1	Smart Watch	100
2	Oven	150

```
SELECT * FROM `product` UNION ALL SELECT * FROM `product2`;
```

UNION ALL Output

productid	productname	productcost
1	Computer	250
2	Television	350
3	Mobile	400
4	Laptop	500
1	Smart Watch	100
2	Oven	150
3	Mobile	400

Q14: What are MySQL Aliases?

MySQL aliases gives temporary name or a nick name to a table or a column these aliases are very useful when writing complex joins or long queries. They make long queries look shorter and column or table name more readable. To give alias name to any table we have to specify "**AS**" after tablename or to any column then "**AS**" after columname.

```
SELECT P.listingid, P.propertyname, L.location FROM `propertylisting` AS P INNER JOIN `location` AS L ON P.locationid = L.id;
```

Q15: Explain MySQL Subquery?

A query inside a query or nested query is called as a sub query

Outer query is called as main query or parent query and inside query is known as child query or inner query.

Subqueries can be used with the SELECT, INSERT, UPDATE, and DELETE statements along with the operators like =, <, >, >=, <=, IN, BETWEEN, etc.

The main reason we use subqueries A subquery returns data that will be used in the main query as a condition to further restrict the data to be retrieved.

Subqueries allow you to use the results of another query in the outer query.

Subqueries divide the complex query into isolated parts so that a complex query can be broken down into a series of logical steps.

It is easy to understand and code maintenance is also at ease.

PropertyListing Table

PropertyID	PropertyName	LocationID
1001	Raheja Exotica	1
1002	Kalpataru Pinnacle	2
1003	Raheja Ridgewood	1
1004	Vasant Oasis	2
1005	Sheth Avante	1

Location Table

id	location
1	India
2	Australia
5	Nepal

```
SELECT * FROM `propertylisting` Where `locatonid` IN (SELECT `id` FROM `location` Where `location` = 'India');
```

Output Table

listingid	propertyname	locatonid
1	Raheja Exotica	1
3	Raheja Ridgewood	1

Q16: Blob vs Text data types in MYSQL?

When we go to MySQL data-types we have one data-type called Blob and another one text.

Associated with Blob data type we have variants depending on sizes like TINYBLOB, MEDIUMBLOG, BLOB, LONGBLOB.

Similarly, we have variants for Text like TINYTEXT, MEDIUMTEXT, TEXT.

So, BLOB stands for Binary Large Objects and as its name suggests, it can be used for storing binary data means we can store pictures, videos, sounds and files etc.

Where else if we want to store large amount string data then we can simply use Text data type.

BLOB does not have a character set

Text has character set.

Q17: CHAR vs VARCHAR data types?

Char and Varchar is a datatype of MySQL which help us to store Character values

and Char is of Fixed Length and where else Varchar is variable length

means suppose if your Char length is 3 and if you store 2 Character values then also Char will take fixed length by adding blank spaces or white spaces to remaining spaces.

On the other side Varchar is flexible and depending on data it changes it size.

Location Table

id	location	chartype
1	India	IND
2	Australia	AUS
5	Nepal	NP

```
SELECT `location`, LENGTH(`location`) AS VarLength, `chartype`, LENGTH(`chartype`) AS CharLength FROM `location`;
```

OUTPUTTable

location	VarLength	chartype	CharLength
India	5	IND	3
Australia	9	AU	2
Nepal	5	NPL	3

Q18: What are the differences between InnoDB and MyISAM engines?

InnoDB	MyISAM
MySQL 8.0 current version: Default storage engine	Prior to MYSQL 5.5: It was default storage engine
It provides the standard ACID-compliant transaction features, along with foreign key support.	It is not ACID complaint and even it does not support relationship constraint.
Supports Transaction Features like Rollback and Commit	No Transaction Feature
Supports Row Level Locking	Supports Table Level Locking
No Full Text Search	Supports Full Text Search

Q19: What are some of the common MySQL commands?

MySQL Commands

```
* Select
* Insert
* update
* Delete
* Create Table
* Drop Table
* Create Database
* Drop Database
* Alter Table
* Create Index
* Drop Index
* Create Trigger
```

Q20: What are aggregate functions in MySQL?

MySQL provides many aggregate functions that are listed below.

```
SUM()
AVG()
COUNT()
MIN()
MAX()
```

Happy Learning, Happy Job Hunting……!