DUMPSARENA

MySQL 8.0 Database Administrator

Oracle 1z0-908

Version Demo

Total Demo Questions: 19

Total Premium Questions: 193

Buy Premium PDF

https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

Topic Break Down

Topic	No. of Questions
Topic 1, MySQL Architecture	19
Topic 2, MySQL Installation	7
Topic 3, MySQL Configuration	14
Topic 4, MySQL Security	42
Topic 5, MySQL Database Management	14
Topic 6, MySQL Backup and Recovery	32
Topic 7, MySQL Optimization	25
Topic 8, MySQL Replication	25
Topic 9, MySQL High Availability	14
Topic 10, Mix Questions	1
Total	193

QUESTION NO: 1

Which two statements are true about using backups of the binary log? (Choose two.)

A. Multiple binary logs can be used to restore data.

B. Multiple binary logs can be applied in parallel for faster data restoration.

C. Binary logs are relatively small, and therefore, excellent for long-term storage and disaster recovery.

D. Binary logs can always be used to unapply unwanted schema changes.

E. They allow for point-in-time recovery of the data.

ANSWER: A E

Explanation:

Multiple binary logs can be used to restore data because the binary log records database changes in chronological order across a sequence of log files. After restoring a full backup, an administrator can use `mysqlbinlog` to process the relevant binary logs, beginning at the appropriate position or timestamp and continuing through subsequent files. This replays the changes made after the backup and brings the restored data forward to the desired state.

They allow for point-in-time recovery of the data because binary logging supplies the change history needed to recover from a backup to a specific moment before an error, such as an accidental update or deletion. The normal recovery workflow restores the latest suitable full backup, then applies binary-log events only up to a selected date and time, log position, or transaction boundary. For this purpose, the required binary logs must still be available and must cover the period from the backup through the recovery target. Oracle's MySQL documentation describes binary logs as essential to point-in-time recovery and documents using `mysqlbinlog` to apply one or more log files during recovery. See MySQL Point-in-Time Recovery and mysqlbinlog — Utility for Processing Binary Log Files.

QUESTION NO: 2

Which condition is true about the use of the hash join algorithm?

A. No index can be used for the join.

B. The query must access no more than two tables.

C. The smallest of the tables in the join must fit in memory as set by join_buffer_size.

D. At least one of the tables in the join must have a hash index.

ANSWER: A

Explanation:

No index can be used for the join. is the applicable condition. A hash join is especially useful when the optimizer has no usable index for evaluating the join predicate efficiently. Instead of locating matching rows through indexed lookups, MySQL builds an in-memory hash table from rows on one input (the build side), using the join key as the hash key. It then reads rows from the other input (the probe side) and uses their join-key values to find candidate matches in that hash table. This avoids repeated scans or inefficient lookup operations where an index cannot support the join condition.

In MySQL 8.0, the optimizer determines whether a hash join is appropriate as part of its cost-based plan selection. The essential point is that the relevant issue is whether an index can be used for the join operation, not whether a table merely has some index defined. MySQL documentation describes hash joins as a join optimization used in cases such as joins lacking indexes on the join columns. See the MySQL 8.0 Reference Manual: Hash Join Optimization and the MySQL 8.0 Optimization Overview.

QUESTION NO: 3

Consider this shell output and executed commands:

[root@oel7 ~]# ps aux | grep mysqld

mysql 2076 3.5 24.6 1386852 372572 ? Ssl 12:01 0:01 /usr/sbin/mysqid [root@oel7 ~]# kill -15 2076

Which statement is true about MySQL server shutdown?

A. kill -15 should be avoided. Use other methods such as mysqladmin shutdown or systemctl stop mysqld.

B. kill -15 and kill -9 are effectively the same forced shutdown that risk committed transactions not written to disk.

C. kill -15 carries out a normal shutdown process, such as mysqladmin shutdown.

D. mysqld_safe prohibits commands that would harm the operation of the server. An error would be returned by the kill command.

ANSWER: C

Explanation:

`kill -15 carries out a normal shutdown process, such as mysqladmin shutdown.` is correct because signal 15 is `SIGTERM`, a termination signal that `mysqld` handles as a request for an orderly shutdown. When MySQL receives this signal, it stops accepting new client connections, terminates or completes active processing as appropriate, shuts down background activity, and performs the storage-engine shutdown work needed to leave data files in a consistent state. For InnoDB, this includes flushing dirty pages and completing the controlled shutdown procedures designed to preserve transactional consistency and support reliable subsequent startup.

This is consistent with MySQL's documented server shutdown behavior: sending `SIGTERM` to the server process is a supported administrative mechanism for initiating a normal shutdown. Commands such as `mysqladmin shutdown` and service-manager actions ultimately request the same type of controlled server termination, although they may use authentication, service scripts, or system management interfaces to do so. The account issuing the operating-system signal must have permission to signal the MySQL server process. See the MySQL Reference Manual: Server Shutdown and MySQL Server Options and System Variables for MySQL server administration details.

QUESTION NO: 4

Which two statements are true about the slow query log? (Choose two.)

A. It can be enabled by setting the global slow_query_log variable.

B. long_query_time determines the execution-time threshold for logging statements.

C. It records every client connection and disconnect.

D. It is required for binary logging.

E. It records only statements that modify data.

ANSWER: A B

Explanation:

The slow query log can be enabled dynamically by setting the global `slow_query_log` system variable. When enabled, it logs qualifying statements based on execution time and other configured conditions.

The `long_query_time` system variable specifies the threshold used to determine whether a statement is considered slow. It can be set with microsecond resolution. The slow query log is not a complete audit log because statements that do not meet its logging conditions are normally omitted.

See The Slow Query Log and long_query_time.

QUESTION NO: 5

A MySQL 8.0 account uses the `caching_sha2_password` authentication plugin.

Which statement is true about this plugin?

A. It is the default authentication plugin in MySQL 8.0.

B. It stores account passwords as plaintext values.

C. It can be used only for local socket connections.

D. It requires the server to disable TLS.

ANSWER: A

Explanation:

`caching_sha2_password` is the default authentication plugin in MySQL 8.0. It provides stronger password authentication than the older `mysql_native_password` mechanism and uses a cache to improve the performance of repeated authentication operations.

Clients must support the authentication protocol used by the plugin. For secure connections, TLS can protect password exchange; otherwise, the plugin supports RSA-based password exchange when the required key material is available.

See Caching SHA-2 Pluggable Authentication.

QUESTION NO: 6

You wish to protect your MySQL database against SQL injection attacks.

Which method would fail to do this?

A. installing and configuring the Connection Control plugin

B. avoiding concatenation of SQL statements and user-supplied values in an application

C. using stored procedures for any database access

D. using PREPARED STATEMENTS

ANSWER: A

Explanation:

Installing and configuring the Connection Control plugin would fail to protect an application from SQL injection attacks because this plugin addresses a different threat: repeated unsuccessful connection attempts. It tracks failed logins by account and source host, then introduces an increasing delay before subsequent connection attempts once configured thresholds are exceeded. This can help reduce the effectiveness of password-guessing and brute-force connection attacks, but it does not inspect application input, alter SQL parsing, parameterize values, or prevent untrusted data from changing the meaning of an SQL statement.

SQL injection occurs when an application constructs a query in a way that lets user-controlled input be interpreted as SQL syntax or commands. Protection must therefore be applied at the application/database-access boundary, where queries and their input values are created and executed. Connection throttling takes place during authentication and does not affect statements issued through an already authenticated database session. Oracle's MySQL documentation describes the plugin's purpose as delaying responses to failed connection attempts, confirming its relevance to login-rate control rather than SQL-injection defense. See MySQL Connection Control Plugins and MySQL Prepared Statements.

QUESTION NO: 7

Which two statements are true about MySQL Installer? (Choose two.)

A. It installs most Oracle MySQL products.

B. It performs product upgrades.

C. It provides only GUI-driven, interactive installations.

D. Manual download of separate product packages is required before installing them through MySQL Installer.

E. It provides a uniform installation wizard across multiple platforms.

ANSWER: A B

Explanation:

MySQL Installer installs most Oracle MySQL products on Microsoft Windows from a single installation framework. Its product catalog includes MySQL Server and commonly deployed companion products and tools, such as MySQL Workbench, MySQL Shell, MySQL Router, connectors, documentation, samples, and examples. It can obtain required packages from the MySQL download repository when using the web installer, while the full installer bundle includes the available packages for offline use. MySQL Installer also manages upgrades: it detects installed MySQL products and can guide the administrator through upgrading eligible products, including MySQL Server, while retaining and updating the relevant product configuration as appropriate. These capabilities make it a central Windows deployment and maintenance utility rather than merely a launcher for separately downloaded packages. Oracle documents the supported product catalog and installation workflows in the MySQL Installer Product Catalog, and describes upgrade procedures in Upgrading MySQL with MySQL Installer.

QUESTION NO: 8

Which statement is true about cold backups?

A. They are backups taken from snapshots of a running database.

B. They are backups taken from OS copy commands.

C. They are good to use if only data structures must be backed up but not log files.

D. They are good to use when many users are online accessing the database.

ANSWER: B

Explanation:

They are backups taken from OS copy commands. A cold backup is a physical backup made while the MySQL Server instance is stopped, so the database files are not changing during the copy. Once the server has been shut down cleanly, the administrator can use operating-system file-copy utilities to copy the relevant database files to a separate backup location. Depending on the storage engine and configuration, this includes the data files and the files necessary for consistent recovery, such as InnoDB system tablespace or redo-log files where applicable. Because no active transactions or writes occur while the server is offline, the copied files represent a consistent state without needing to coordinate concurrent database activity.

MySQL documentation categorizes this as a physical backup approach: files and directories containing database contents are copied directly, rather than exporting SQL statements. The defining characteristic of a cold backup is that the server is not running during the file copy; the operating-system copy operation is the mechanism used to produce that offline physical backup. See the MySQL Reference Manual discussion of backup types and its guidance on backup methods.

QUESTION NO: 9

User account baduser@hostname on your MySQL instance has been compromised.

Which two commands stop any new connections using the compromised account? (Choose two.)

A. ALTER USER baduser@hostname PASSWORD DISABLED;

B. ALTER USER baduser@hostname MAX_USER_CONNECTIONS 0;

C. ALTER USER baduser@hostname ACCOUNT LOCK;

D. ALTER USER baduser@hostname IDENTIFIED WITH mysql_no_login;

E. ALTER USER baduser@hostname DEFAULT ROLE NONE;

ANSWER: C D

Explanation:

`ALTER USER baduser@hostname ACCOUNT LOCK;` is correct because MySQL account locking prevents new client connections for that account immediately. The lock state is stored as an account attribute, and it remains in effect until an administrator explicitly issues `ALTER USER ... ACCOUNT UNLOCK`. This is an appropriate immediate containment action when credentials may have been exposed. MySQL documents account locking as part of the `ALTER USER` statement: MySQL 8.0 ALTER USER Statement.

`ALTER USER baduser@hostname IDENTIFIED WITH mysql_no_login;` is also correct. The `mysql_no_login` authentication plugin intentionally does not permit authentication by client connections. Assigning it to the compromised account ensures that subsequent attempts to connect as that account cannot authenticate, providing another direct method to block new use of the account. This plugin is specifically intended for accounts that must not be used to log in, including internal or otherwise restricted accounts. Its behavior and intended use are described in the MySQL No-Login Pluggable Authentication documentation.

QUESTION NO: 10

Examine these InnoDB Cluster parameter settings:

```
cluster.setInstanceOption('host1:3377', 'memberWeight', 40)
cluster.setInstanceOption('host2:3377', 'memberWeight', 30)
cluster.setInstanceOption('host3:3377', 'memberWeight', 40)
cluster.setInstanceOption('host3:3377', 'exitStateAction', "ABORT_SERVER")
cluster.setOption("expelTimeout", 1)
```

Now examine the partial status:



A permanent network failure isolates host3.

Which two statements are true? (Choose two.)

A. The instance deployed on host2 is elected as the new primary instance.

B. The instance deployed on host3 is expelled from the cluster and must be rejoined using cluster.addInstance (' host3:3377 ')

C. The instance deployed on host3 will automatically rejoin the cluster when connectivity is re-established.

D. Failure of the instance deployed on host1 provokes an outage.

E. The issuing command cluster.switchToMultiPrimaryMode() will fail to enable multi-primary mode.

F. The primary instance can be specified by using the command cluster.setPrimaryInstance(':').

ANSWER: C F

Explanation:

The instance deployed on host3 will automatically rejoin the cluster when connectivity is re-established because the Cluster configuration enables automatic rejoin attempts. When Group Replication expels a member after it is unreachable for the configured period, MySQL Server can use the configured automatic-rejoin setting to attempt to reconnect and recover its membership. Once host3 can again communicate with a viable majority of the group, it performs distributed recovery as needed and returns to the cluster without an administrator manually issuing a rejoin operation, provided connectivity returns while the configured retry attempts remain available.

The primary instance can be specified by using the command `cluster.setPrimaryInstance('<host>:<port>')` because the displayed Cluster is operating in single-primary mode. MySQL Shell's `Cluster.setPrimaryInstance()` operation selects an ONLINE cluster member as the new primary and safely coordinates the primary-role change through Group Replication. The target is supplied as an instance connection definition such as `'host2:3306'`. This allows an administrator to choose the writable primary explicitly rather than waiting for an automatic election caused by loss of the current primary. See the MySQL documentation for automatic Cluster rejoin and managing the InnoDB Cluster primary instance.

QUESTION NO: 11

You are taking a physical backup using a backup tool that supports backup locks.

Which statement is true about this command?

```
mysql> LOCK INSTANCE FOR BACKUP;
```

A. It permits DML while blocking operations that can compromise a physical backup.

B. It blocks all SELECT, INSERT, UPDATE, and DELETE statements.

C. It permanently disables DDL for the instance.

D. It can be acquired only when binary logging is disabled.

ANSWER: A

Explanation:

`LOCK INSTANCE FOR BACKUP` permits normal DML activity while preventing operations that could interfere with a physical backup, such as operations that modify files required by the backup. This provides a less disruptive alternative to a global read lock for backup operations that require protection against incompatible DDL and file-level changes.

The lock is released with `UNLOCK INSTANCE` or when the session holding it ends. The account requires the `BACKUP_ADMIN` privilege to acquire the lock. See LOCK INSTANCE FOR BACKUP and UNLOCK INSTANCE Statements.

QUESTION NO: 12

Which three sets of item information are visible in the mysql system database? (Choose three.)

A. help topics

B. performance monitoring information

C. plugins

D. information about table structures

E. audit log events

F. rollback segments

G. time zone information and definitions

ANSWER: A C G

Explanation:

The `mysql` system schema contains tables used by the MySQL server to store operational metadata and system-maintained information. It includes the server's help-system tables, such as `help_topic`, `help_category`, `help_keyword`, and `help_relation`. These tables provide the data behind MySQL's built-in help capability, including the information returned by `HELP` statements. Therefore, help topics are visible in this schema.

Plugin registration and status metadata is also maintained through the `mysql.plugin` table. This table records installed server plugins that should be loaded by the server, making plugins another set of information visible in the `mysql` database.

In addition, MySQL stores named time-zone definitions in the `mysql` schema. The time-zone tables include `time_zone`, `time_zone_name`, `time_zone_transition`, and related tables. These are populated when time-zone data is loaded and enable support for named time zones and time-zone conversions. Oracle's documentation lists these help, plugin, and time-zone tables as part of the MySQL system schema. See the MySQL System Schema documentation and the time-zone table loading documentation.

QUESTION NO: 13

Your MySQL instance is capturing a huge amount of financial transactions every day in the finance database.

Company policy is to create a backup every day.

The main tables being updated are prefixed with transactions-.

These tables are archived into tables that are prefixed with archives- each month. mysqlbackup --optimistic-busy-tables="^finance\.transactions-.*" backup

Which optimization process best describes what happens with the redo logs?

A. The redo logs are backed up first, then the transaction and archive tables.

B. The redo logs are backed up only if there are changes showing for the transactions tables.

C. The redo logs are not backed up at all.

D. The archive tables are backed up first, then the transaction tables and redo logs.

E. The transaction tables are backed up first, then the archive tables and redo logs.

ANSWER: B

Explanation:

The redo logs are backed up only if there are changes showing for the transactions tables. The `--optimistic-busy-tables` option is part of MySQL Enterprise Backup's optimistic backup processing. It identifies tables that are expected to be busy, using a regular expression. Here, `^finance\.transactions-.*` matches tables in the `finance` schema whose names begin with `transactions-`. This lets MySQL Enterprise Backup focus its change-handling work on the specified high-update tables rather than treating every table as equally likely to require redo processing.

Optimistic backup reduces unnecessary redo-log work by initially backing up data optimistically and retaining redo information only when it is needed to account for modifications detected for the designated busy tables. Because the transaction tables are the continuously updated objects in this scenario, redo logging is captured as required to preserve a consistent, recoverable backup when changes are found. The monthly archive tables do not match the supplied regular expression and are not the tables designated for this busy-table optimization. This approach is useful when a known subset of tables receives most write activity, because it avoids imposing the same redo-log overhead for tables that are not changing in the relevant way during the backup. See the MySQL Enterprise Backup documentation on optimistic backup and the mysqlbackup command options.

QUESTION NO: 14

Which two tools are available to monitor the global status of InnoDB locking? (Choose two.)

A. SHOW ENGINE INNODB STATUS;

B. INFORMATION_SCHEMA.INNODB_METRICS

C. SHOW TABLE STATUS;

D. INFORMATION_SCHEMA.STATISTICS

E. INFORMATION_SCHEMA.INNODB_TABLESTATS

F. SHOW STATUS;

ANSWER: A F

Explanation:

`SHOW ENGINE INNODB STATUS;` provides the InnoDB Monitor report, including the TRANSACTIONS section. This output gives a point-in-time, server-wide view of InnoDB transaction activity, lock waits, waiting transactions, and relevant lock details. It is a primary diagnostic tool when investigating active contention or deadlocks. Oracle documents the locking and transaction information available through this statement in the InnoDB Standard Monitor documentation.

`SHOW STATUS;` exposes global status counters that summarize InnoDB row-lock activity for the server. The `Innodb_row_lock_current_waits`, `Innodb_row_lock_time`, `Innodb_row_lock_time_avg`, `Innodb_row_lock_time_max`, and `Innodb_row_lock_waits` status variables enable ongoing monitoring of lock waits and their cumulative impact. These counters are especially useful for periodic sampling, dashboards, and identifying trends in contention across the instance rather than inspecting one individual session. The meanings and availability of these server status variables are listed in the MySQL Server Status Variables reference.

QUESTION NO: 15

Examine this MySQL Shell command:

dba.rebootClusterFromCompleteOutage()

Which two statements are true? (Choose two.)

A. It reconfigures InnoDB Cluster if the cluster was stopped.

B. It performs InnoDB Cluster instances rolling restart.

C. It only starts all InnoDB Cluster instances.

D. It is not mandatory that all instances are running and reachable before running the command.

E. It stops and restarts all InnoDB Cluster instances and initializes the metadata.

F. It only stops and restarts all InnoDB Cluster instances.

G. It picks the minimum number of instances necessary to rebuild the quorum and reconfigures InnoDB Cluster.

ANSWER: A D

Explanation:

`dba.rebootClusterFromCompleteOutage()` is the AdminAPI recovery operation for an InnoDB Cluster that has suffered a complete outage, meaning Group Replication is no longer operating on any cluster member. It reconfigures and starts Group Replication again, selecting an appropriate instance to seed the recovered cluster. MySQL Shell evaluates the members' GTID execution state to help ensure that the most up-to-date available member is used as the recovery seed, then the other available members can rejoin the cluster.

It is also not inherently necessary for every member to be running before the operation begins; a complete outage commonly includes stopped or offline Group Replication members. The command supports recovery where members are unavailable, subject to the command's reachability checks and its `force` option. Using `force` permits the reboot process to continue when some configured instances cannot be reached, although this should be used carefully because unreachable members might contain transactions that must be considered before recovery.

Oracle documents this operation as the mechanism to restore an InnoDB Cluster after all members are offline and describes its validation and recovery behavior in the MySQL Shell AdminAPI reference: Rebooting a Cluster from a Complete Outage and MySQL InnoDB Cluster.

QUESTION NO: 16

Examine this statement, which executes successfully:

Now examine this query:

Which two statements can do this? (Choose two.)

A. ADD INDEX (birth_date DESC);

B. ADD INDEX ((MONTH(birth_date)));

C. ADD COLUMN birth_month tinyint unsigned GENERATED ALWAYS AS (MONTH(birth_date)) VIRTUAL NOT NULL,

D. ADD COLUMN birth_month tinyint unsigned GENERATED ALWAYS AS (birth_date->>'$.month') VIRTUAL NOT NULL,

E. ADD INDEX ((CAST(birth_date->>'$.month' AS unsigned)));

F. ADD INDEX (birth_date);

ANSWER: A F

Explanation:

`ADD INDEX (birth_date DESC);` creates a descending index key part for `birth_date`. MySQL can use this index to retrieve rows in descending `birth_date` order without a separate filesort when the query's ordering and other access conditions permit index-order retrieval. Descending key parts are particularly important for composite indexes whose requested sort directions cannot all be satisfied merely by scanning an ordinary index in reverse.

`ADD INDEX (birth_date);` is also suitable for a single-column descending ordering requirement. MySQL B-tree indexes can be scanned in either the forward or backward direction. Therefore, an ascending single-column index on `birth_date` can be scanned backward to produce rows ordered by `birth_date DESC`. The optimizer determines whether using either index is less costly than alternative access paths, but both definitions provide an index order that can satisfy a descending sort on that one column. Oracle's documentation describes backward scans and descending indexes in the Descending Indexes section and details index-based ordering in ORDER BY Optimization.

QUESTION NO: 17

Examine this partial output for InnoDB Cluster status:



```
"topology": {
   "host1:3377": {
      "address": "host1:3377",
      "mode": "R/W",
      [...]
      "status": "ONLINE",
      "version": "8.0.18"
   },
   "host2:3377": {
      "address": "host2:3377",
      "mode": "R/O",
      [...]
      "status": "(MISSING)"
   },
   "host3:3377": {
      "address": "host3:3377",
      "mode": "R/O",
      [...]
      "status": "ONLINE",
      "version": "8.0.18"
   }
}
```

Which statement explains the state of the instance deployed on host2?

A. It can be recovered from a donor instance on host3 by cloning using the command cluster.rejoinInstance ('@host3:3377')

B. It can rejoin the cluster by using the command cluster.addInstance('@host3:3377')

C. It has been removed from the cluster by using the command STOP GROUP_REPLICATION;

D. It can rejoin the cluster by using the command dba.rebootClusterFromCompleteOutage()

E. It has been expelled from the cluster because of a transaction error.

ANSWER: D

Explanation:

It can rejoin the cluster by using the command `dba.rebootClusterFromCompleteOutage()`. The status shown is consistent with an InnoDB Cluster complete outage: Group Replication is no longer running as an active group, so normal automatic membership recovery is not sufficient to restore cluster operation. MySQL Shell provides `dba.rebootClusterFromCompleteOutage()` specifically for this recovery scenario. The operation identifies the most appropriate instance from which to restore the cluster's Group Replication topology, starts Group Replication on that instance, and then brings the remaining instances back into the cluster as appropriate. Before performing the reboot, the administrator should ensure that the selected seed instance contains the required transaction history and should resolve any data consistency concerns, because a complete outage can leave members with differing executed transaction sets. The recovery should be performed through MySQL Shell against a reachable cluster member and with credentials authorized to administer the instances. Oracle documents this procedure as the supported way to recover an InnoDB Cluster following a complete Group Replication outage. See the MySQL Shell recovery documentation and the InnoDB Cluster documentation.

QUESTION NO: 18

You want to check the values of the sort_buffer_size session variables of all existing connections.

Which performance_schema table can you query?

A. user_variables_by_thread

B. global_variables

C. variables_by_thread

D. session_variables

ANSWER: C

Explanation:

`variables_by_thread` is the appropriate Performance Schema table because it exposes system-variable values on a per-thread basis. A client connection is represented by a foreground thread in Performance Schema, so querying this table lets an administrator inspect the session-specific value of `sort_buffer_size` for each existing connection. The table includes identifiers such as `THREAD_ID`, the variable name, and its value, allowing the results to be joined to Performance Schema thread metadata when connection details are also needed.

This is particularly useful because `sort_buffer_size` is a session system variable: each connection can retain its own effective value, including a value set with a session-level `SET` statement. For example, filtering `variables_by_thread` for `VARIABLE_NAME = 'sort_buffer_size'` returns the relevant setting for every instrumented thread represented in the table. MySQL documents that `variables_by_thread` contains session system-variable values for each thread and identifies the thread to which each value applies. See the MySQL Performance Schema variables_by_thread Table documentation and the sort_buffer_size system-variable reference.

QUESTION NO: 19

You want to dump all databases with names that start with "db".

Which command will achieve this?

A. mysqlpump --include-tables=db.% --result-file=all_db_backup.sql

B. mysqlpump > all_db_backup.sql

C. mysqlpump --include-databases=db --result-file=all_db_backup.sql

D. mysqlpump --include-databases=db% --result-file=all_db_backup.sql

ANSWER: D

Explanation:

`mysqlpump --include-databases=db% --result-file=all_db_backup.sql` is correct because `--include-databases` accepts database-name patterns for object selection. In mysqlpump patterns, the percent sign (`%`) is a wildcard matching any sequence of characters. Therefore, the pattern `db%` selects every database whose name begins with `db`, such as `db`, `db1`, and `db_sales`. The selected databases and their included objects are written to the file specified by `--result-file=all_db_backup.sql`, rather than being sent to standard output. This makes the command suitable for creating one SQL logical-backup file containing all matching databases. mysqlpump supports include and exclude options to control the objects included in a dump, and database patterns provide the required prefix-based selection behavior here. See the MySQL Reference Manual documentation for mysqlpump and its --include-databases option.