

SHOW DATABASES; – Lists all databases.
USE database_name; – Switches to a specific database.
SHOW TABLES; – Lists all tables in the current database.
DESCRIBE table_name; – Shows the table structure.
SHOW INDEX FROM table_name; – Displays table indexes.
SHOW TABLE STATUS LIKE 'table_name'; – Gives table info (size, creation date).
CHECK TABLE table_name; – Checks for errors in the table.
ANALYZE TABLE table_name; – Analyzes index distribution.
OPTIMIZE TABLE table_name; – Reclaims unused space.
SELECT * FROM table_name; – Selects all rows.
SELECT * FROM table_name WHERE column = 'value'; – Filters by value.
SELECT * FROM table_name LIKE '%pattern%'; – Finds pattern matches.
SELECT * FROM table_name ORDER BY column ASC|DESC; – Sorts by column.
SELECT * FROM table_name LIMIT 10; – Limits to 10 rows.
SELECT * FROM table_name WHERE column IS NOT NULL; – Filters non-null values.
SELECT * FROM table_name WHERE column BETWEEN x AND y; – Filters by range.
SELECT * FROM table_name GROUP BY column; – Groups by column.
SELECT * FROM table_name HAVING COUNT(column) > 1; – Filters by group count.
SELECT * FROM information_schema.SCHEMATA WHERE schema_name = 'database_name'; – Shows database metadata.
SELECT * FROM table_name INNER JOIN another_table ON table.id = another.id; – Joins two tables.

mysqldump -u user -p database_name > backup.sql – Exports (dumps) a database to an SQL file.
mysqldump -u user -p --all-databases > all_databases_backup.sql – Dumps all databases to a single SQL file.
mysqldump -u user -p --databases database_name > all_databases_backup.sql – erstellt auch die datenbank
mysql -u user -p database_name < backup.sql – Imports an SQL file into a database.
source /path/to/backup.sql – Imports an SQL file directly from the MySQL CLI.
mysqlimport -u user -p database_name /path/to/file.csv – Imports data from a CSV file into a table.
GRANT SELECT ON database.* TO 'username'@'host'; – Allows reading data.
GRANT INSERT ON database.* TO 'username'@'host'; – Allows inserting data.
GRANT UPDATE ON database.* TO 'username'@'host'; – Allows updating data.
GRANT DELETE ON database.* TO 'username'@'host'; – Allows deleting data.
GRANT CREATE ON database.* TO 'username'@'host'; – Allows creating tables and databases.
GRANT DROP ON database.* TO 'username'@'host'; – Allows dropping tables and databases.
GRANT ALTER ON database.* TO 'username'@'host'; – Allows altering tables (e.g., adding/removing columns).
GRANT INDEX ON database.* TO 'username'@'host'; – Allows creating and removing indexes.
GRANT EXECUTE ON database.* TO 'username'@'host'; – Allows executing stored procedures and functions.
GRANT CREATE VIEW ON database.* TO 'username'@'host'; – Allows creating views.
GRANT SHOW VIEW ON database.* TO 'username'@'host'; – Allows viewing the definition of views.
GRANT CREATE ROUTINE ON database.* TO 'username'@'host'; – Allows creating stored procedures/functions.
GRANT ALTER ROUTINE ON database.* TO 'username'@'host'; – Allows altering stored procedures/functions.
GRANT EVENT ON database.* TO 'username'@'host'; – Allows creating events for scheduling.
GRANT TRIGGER ON database.* TO 'username'@'host'; – Allows creating triggers.
GRANT ALL PRIVILEGES ON database.* TO 'username'@'host'; – Allows creating triggers.

1.) Zeige auf, welche Stakeholder von den User Stories angesprochen werden.
Stakeholder 1Anwender
Stakeholder 2Entwickler
2.) Benenne mindestens 5 weitere mögliche Stakeholder, die in Projekten vorkommen können.
Stakeholder 3Kunde
Stakeholder 4Auftraggeber
Stakeholder 5Projektleiter
Stakeholder 6Gesetzgeber
Stakeholder 7Mitbewerber

3.) In welchen Dokumentationen findest du Inhalte aus den User Stories wieder? Bsp.; Testkonzept
Dokument 1 Lasten- / Pflichtenheft
Dokument 2 Anwenderdokumentation
Dokument 3 Detailstudie / Konzept
4.) Welchen Einfluss haben die drei User Stories auf das Mengengerüst bzw. welche Informationen fehlen?
Beispiel: Welche Medien sollen verwaltet werden können?
Information 1 Welche Datenquellen können verwendet werden ? (IMDB/TMDB/etc.)
Information 2 Wie viele Medien sollen verwaltet werden?
Information 3 Müssen mehrere Benutzer die Medien verwalten können?

Documentation MySQL Datenbank:
Apex SQL Doc für MySQL (Trial)

User-Story 1

WER	Als Anwender möchte ich...
ZIEL	alle meine digitalisierten Medien verwalten können,
NUTZEN	damit ich jederzeit gesuchte Daten rasch finden kann.

Service Status einsehen:

Systemctl status mysql

Service starten und stoppen:

Systemctl stop mysql
Systemctl start mysql

Connection Checks:

mysqladmin -u root -p variables

mysqladmin -u root -p version

mysqlshow -u root -p # zeigt datenbanken

Technik	Vorteile	Nachteile
Einzel-Interview	Effizient Interaktiv Beugt Missverständnissen vor	Gute Vorkenntnisse erforderlich Auswahl der Gesprächspartner nicht immer einfach Terminplanung erforderlich
Gruppen-Interview	Effizient Interaktiv Beugt Missverständnissen vor Ermöglicht Diskussionen	Gute Vorkenntnisse erforderlich Auswahl der Gesprächspartner nicht immer einfach Terminplanung erforderlich Moderationskenntnisse wünschenswert
Umfrage	Gleichzeitig viele Personen ansprechen Breit abgestützte Ergebnisse Zeit und Ort unabhängig	Grosse Vorbereitung für Frage Katalog erforderlich Missverständnisse können Ergebnisse verfälschen
User Story	Einfache und verständliche Sprache Nachvollziehbare Beschreibung	Unterschiedliche Beschreibungen müssen konsolidiert werden

Testing
SHOW WARNINGS; – Lists warnings from the last statement.
SELECT BENCHMARK(1000000, EXP(10)); – Tests server performance.
EXPLAIN SELECT * FROM table_name; – Analyzes query execution.
CHECK TABLE table_name; – Tests table integrity.
SHOW ENGINE INNODB STATUS; – Shows InnoDB diagnostics.
SET sql_mode = 'STRICT_ALL_TABLES'; – Tests data integrity in strict mode.
SHOW PROCESSLIST; – Displays active queries and connections.
mysqlslap -u root -p --concurrency=5--iterations=20--number-int-cols=2--number-char-cols=3--auto-generate-sql
mysqladmin -u root -p version

Die Index Typen sind:
Primary
Hiermit wird der Primary Key festgelegt, dieser ist erstens unique als auch die vorherrschende Art, wie auf die Tabelle zugegriffen wird.
Key oder Index
Dient nur zur Optimierung. Es können also mehrere Treffer für ein Indexattribut gefunden werden.
Fulltext
Dient dazu Textfelder auch mit like „%abx%“ effizient abzufragen.
Unique
Für Felder(oder Kombinationen), die nur einmal vorkommen dürfen.
Spatial
Ist Spezialisiert für Geo Daten (Höhe Breite Länge) also in erster Linie für GMS Daten.

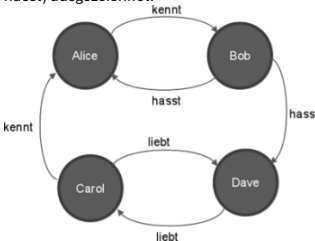
Wie viele Indexe unterstützt MySQL pro Tabelle?
Alle Storage Engines mindestens 16, diverse können mehr:
MyISAM
maximum indexes per table 64
maximum key length 1000 bytes
InnoDB
maximum indexes per table 64
maximum key length 767 bytes

Erstelle mit mysqldump ein **Backup** für folgende Szenarien:

Die Tabelle "albums" der Datenbank "musiccollection" soll gesichert werden (inkl. Daten)
mysqldump –u root –p musiccollection albums > albums.sql
Die Tabelle "genres" der Datenbank "musiccollection" soll gesichert werden (OHNE Daten) mysqldump –u root –p --no-data musiccollection genres> genres.sql
Die Datenbank "musiccollection" soll gesichert werden (inkl. Daten) mysqldump –u root –p musiccollection > musiccollection_bak.sql
Alle Datenbanken sollen gesichert werden **mysqldump –u root –p --all-databases > full_backup.sql**

Jetzt geht's ums Ganze. Das Backup soll nach einem Totalausfall wieder eingespielt werden. Gehe wie folgt vor:

Erstelle ein Backup der Datenbank "musiccollection" inkl. Daten mysqldump –u root –p musiccollection > musiccollection_bak.sql
Lösche die Datenbank musiccollection (DROP Database ...) DROP DATABASE musiccollection;
Spiele das Backup wieder ein **mysql –u root –p < musiccollection_bak.sql**

Datenbank Typ	Eigenschaft																																																																																																										
Hierarchische Datenbank	Mit dem Wurzelknoten lädt man sämtliche abhängige Knoten in den Speicher. Dadurch wird das durchlaufen der Knoten extrem schnell. Ein typischer Vertreter einer Hierarchischen Datenbank ist IMS von IBM.																																																																																																										
Netzwerk Datenbank	Eine Netzwerkdatenbank ist eine Weiterentwicklung der hierarchischen Datenbank. Allerdings sind hier beliebige Verknüpfungen zu anderen Objekten möglich. Eine Netzwerk Datenbank ist zwar sehr flexibel, doch gerade bei vielen Verknüpfungen ist es schwer den Überblick zu behalten. IDMS der Firma CA ist wohl der bekannteste Vertreter.																																																																																																										
Relationale Datenbank	Die relationale Datenbank wurde 1970 von E.F.Codd entwickelt und ist heute der häufigste verwendete Datenbanktyp. Die Daten werden in Tabellen (Relation) gehalten. Die Zeilen der Datenbank sind die einzelnen Objekte, die Spalten nennt man Attribute (Felder). Typische Vertreter einer relationalen Datenbank sind: MYSQL (heute von Oracle), Oracle DB, DB2, SQL-Server (heute microsoft), Ingres																																																																																																										
Objektorientierte Datenbank	Aus dem objektorientierten Ansatz (Java, C++) wurden auch Datenbanken entwickelt. Diese können komplexe Strukturen speichern. Auch wenn die Handhabung leicht zu verstehen ist (wenn man einmal das objektorientierte Konzept verstanden hat) leidet doch oft die Performance.																																																																																																										
Graphendatenbank	<div><div><p>Ein einfaches Beispiel für einen Graphen sind Beziehungen zwischen Menschen (siehe dazu auch Soziogramm). Die Knoten repräsentieren Menschen; jedem Knoten wird dabei der Name der Person zugeordnet. Die Kanten repräsentieren Beziehungen; sie sind durch einen Typ (<i>kennt</i>, <i>liebt</i>, <i>hasst</i>) ausgezeichnet.</p></div><div><table><tr><th>ID / Bezeichnung</th><th>T 001</th><th>Referenz zu Anforderung & Abnahmekriterium</th><th>nn</th></tr><tr><td>Beschreibung</td><td></td><td></td><td>z.B. die Qualitätsanforderungen im Ergebnis Systemanforderungen</td></tr><tr><td>Testvoraussetzung</td><td></td><td></td><td></td></tr><tr><td>Testschritte</td><td></td><td></td><td></td></tr><tr><td>Erwartetes Ergebnis</td><td></td><td></td><td></td></tr></table><table><tr><th>Nr.</th><th>Test-ID</th><th>Ausgeführt von</th><th>Erfolgreich</th><th>Termin</th></tr><tr><td>1</td><td></td><td></td><td></td><td></td></tr><tr><td>2</td><td></td><td></td><td></td><td></td></tr><tr><td>3</td><td></td><td></td><td></td><td></td></tr></table></div></div>	ID / Bezeichnung	T 001	Referenz zu Anforderung & Abnahmekriterium	nn	Beschreibung			z.B. die Qualitätsanforderungen im Ergebnis Systemanforderungen	Testvoraussetzung				Testschritte				Erwartetes Ergebnis				Nr.	Test-ID	Ausgeführt von	Erfolgreich	Termin	1					2					3					<div><p>Welche Angaben in der Ausgabe eines EXPLAIN sind am wichtigsten, um zu erkennen, ob ein SQL Befehl einen Index nutzt? Table, Key Welcher der folgenden SQL DML Befehle kann von einem Index profitieren, welcher nicht? Gelten spezielle Voraussetzungen? SELECT Bei Fremdschlüsseln und WHERE Klauseln INSERT Profitiert gar nicht, im Gegenteil, jeder Index der Tabelle muss noch zusätzlich ergänzt werden UPDATE Nur, wenn mit WHERE Klausel genau Datensätze ausgewählt werden DELETE Nur, wenn mit WHERE Klausel genau Datensätze ausgewählt werden Wenn Sie folgenden Befehl ausführen, was beobachten Sie bezüglich des Speicherplatzbedarfs der Tabelle index_demo? <code>ALTER TABLE `index_demo` ADD INDEX(`a`);</code> Beide Indexe zusammen sind fast so gross wie die eigentlichen Daten der Tabelle selbst. Was bewirkt der Befehl „ANALYZE TABLE“ und wann sollte er eingesetzt werden? Analysiert und speichert die Verteilung der Schlüssel der Indexe einer Tabelle. MySQL nimmt eine gleichmässige Verteilung der Schlüssel an. Eine Analyse kann den Index optimieren. Was bewirkt der Befehl „OPTIMIZE TABLE“ und wann sollte er eingesetzt werden? Optimiert die Datenorganisation der Tabelle, so das Lese-Schreibzugriff schneller werden sollten, ebenfalls Zugriffe auf den Index. Nach ausgiebigem Einfügen und Löschoperationen kann es sinnvoll sein.</p></div> <div><div><div>CREATE USER 'username'@'host' IDENTIFIED BY 'password'; – Creates a new user. RENAME USER 'old_name'@'host' TO 'new_name'@'host'; – Renames a user. SET PASSWORD FOR 'username'@'host' = 'new_password'; – Changes a user's password. GRANT ALL PRIVILEGES ON database.* TO 'username'@'host'; – Grants privileges to a user. REVOKE ALL PRIVILEGES ON database.* FROM 'username'@'host'; – Removes user privileges. SHOW GRANTS FOR 'username'@'host'; – Displays a user's privileges. DROP USER 'username'@'host'; – Deletes a user. FLUSH PRIVILEGES; – Applies privilege changes.</div><table><tr><td>ID</td><td>A1</td><td>A2</td><td>A3</td><td>A4</td><td>A5</td><td>A6</td><td>A7</td><td>A8</td><td>A9</td><td>11</td><td>12</td><td>13</td></tr><tr><td></td><td>TAB1</td><td>TAB2</td><td>TAB3</td><td>TAB4</td><td>LINK</td><td>TAB5</td><td>TAB6</td><td>TAB7</td><td>TAB8</td><td>Art = Aktion</td><td>Art = Aktion</td><td>Art = Aktion</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>Ar = Archivierung</td><td>Ar = Archivierung</td><td>Ar = Archivierung</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>31</td><td>32</td><td>33</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>Benutzer</td><td>Benutzer</td><td>Benutzer</td></tr></table></div></div>	ID	A1	A2	A3	A4	A5	A6	A7	A8	A9	11	12	13		TAB1	TAB2	TAB3	TAB4	LINK	TAB5	TAB6	TAB7	TAB8	Art = Aktion	Art = Aktion	Art = Aktion											Ar = Archivierung	Ar = Archivierung	Ar = Archivierung											31	32	33											Benutzer	Benutzer	Benutzer
ID / Bezeichnung	T 001	Referenz zu Anforderung & Abnahmekriterium	nn																																																																																																								
Beschreibung			z.B. die Qualitätsanforderungen im Ergebnis Systemanforderungen																																																																																																								
Testvoraussetzung																																																																																																											
Testschritte																																																																																																											
Erwartetes Ergebnis																																																																																																											
Nr.	Test-ID	Ausgeführt von	Erfolgreich	Termin																																																																																																							
1																																																																																																											
2																																																																																																											
3																																																																																																											
ID	A1	A2	A3	A4	A5	A6	A7	A8	A9	11	12	13																																																																																															
	TAB1	TAB2	TAB3	TAB4	LINK	TAB5	TAB6	TAB7	TAB8	Art = Aktion	Art = Aktion	Art = Aktion																																																																																															
										Ar = Archivierung	Ar = Archivierung	Ar = Archivierung																																																																																															
										31	32	33																																																																																															
										Benutzer	Benutzer	Benutzer																																																																																															
nosql Datenbanken	Nosql steht für Not only SQL. Kehrt sich also von den rein relationalen ab. NoSql Datenbanken werden vor allem dort eingesetzt, wo rein relationale Datenbanken schwächeln; Indexierung grosser Dokumente, Streaming Medien oder bei grossem Lastaufkommen häufig besuchter Webseiten. Beispiele sind: Apache Jackrabbit, BaseX, CouchDB, MongoDB, IBM Notes ua.																																																																																																										
JSON Strukturen Properties	Das sind in erster Linie embedded Datenbanken, also Datenbanken, denen man apriori nicht ansieht, dass es eine Datenbank ist, sondern sich als Applikationen zeigen. Beispielweise Adressliste der Schweiz auf CD, Kontakte in Handys.																																																																																																										

Welche Angaben in der Ausgabe eines EXPLAIN sind am wichtigsten, um zu erkennen, ob ein SQL Befehl einen Index nutzt?
Table, Key
Welcher der folgenden SQL DML Befehle kann von einem Index profitieren, welcher nicht? Gelten spezielle Voraussetzungen?
SELECT Bei Fremdschlüsseln und WHERE Klauseln
INSERT Profitiert gar nicht, im Gegenteil, jeder Index der Tabelle muss noch zusätzlich ergänzt werden
UPDATE Nur, wenn mit WHERE Klausel genau Datensätze ausgewählt werden
DELETE Nur, wenn mit WHERE Klausel genau Datensätze ausgewählt werden
Wenn Sie folgenden Befehl ausführen, was beobachten Sie bezüglich des Speicherplatzbedarfs der Tabelle index_demo? ALTER TABLE `index_demo` ADD INDEX(`a`);
Beide Indexe zusammen sind fast so gross wie die eigentlichen Daten der Tabelle selbst.
Was bewirkt der Befehl „ANALYZE TABLE“ und wann sollte er eingesetzt werden?
Analysiert und speichert die Verteilung der Schlüssel der Indexe einer Tabelle. MySQL nimmt eine gleichmässige Verteilung der Schlüssel an. Eine Analyse kann den Index optimieren.
Was bewirkt der Befehl „OPTIMIZE TABLE“ und wann sollte er eingesetzt werden?
Optimiert die Datenorganisation der Tabelle, so das Lese-Schreibzugriff schneller werden sollten, ebenfalls Zugriffe auf den Index.
Nach ausgiebigem Einfügen und Löschooperationen kann es sinnvoll sein.

CREATE USER 'username'@'host' IDENTIFIED BY 'password'; – Creates a new user.
RENAME USER 'old_name'@'host' TO 'new_name'@'host'; – Renames a user.
SET PASSWORD FOR 'username'@'host' = 'new_password'; – Changes a user's password.
GRANT ALL PRIVILEGES ON database.* TO 'username'@'host'; – Grants privileges to a user.
REVOKE ALL PRIVILEGES ON database.* FROM 'username'@'host'; – Removes user privileges.
SHOW GRANTS FOR 'username'@'host'; – Displays a user's privileges.
DROP USER 'username'@'host'; – Deletes a user.
FLUSH PRIVILEGES; – Applies privilege changes.

EXPLAIN SELECT * FROM table_name; – Analyzes query performance and suggests indexes.
SHOW STATUS LIKE 'Handler%'; – Displays status counters for table handler operations.
SHOW VARIABLES LIKE 'query_cache%'; – Shows query cache settings for optimizing repeated queries.
SHOW INDEX FROM table_name; – Lists indexes on a table to identify optimization opportunities.
OPTIMIZE TABLE table_name; – Reorganizes a table to reclaim unused space and improve efficiency.
ANALYZE TABLE table_name; – Updates statistics for indexes to improve query plans.
ALTER TABLE table_name ENGINE = InnoDB; – Converts a table to InnoDB for better performance on transactional workloads.
SET profiling = 1; – Enables profiling for tracking query execution times.
SHOW PROFILES; – Lists recent query profiles to identify slow queries.
SHOW PROFILE FOR QUERY query_id; – Displays detailed performance data for a specific query by ID.

Datenbanktyp	Vorteile	Nachteile
Hierarchisches Datenbankmodell	<ul style="list-style-type: none">einfache Strukturschneller ZugriffDatenintegrität und Datenunabhängigkeit bleiben erhaltenauch für grosse Datenmengen geeignet	<ul style="list-style-type: none">setzt Kenntnisse der Struktur vorausjede Beziehung erfordert eigene Definitionpro Satz nur ein Feld und eine Verknüpfungnachträgliche Strukturänderungen kaum möglich
Netzartiges Datenbankmodell	<ul style="list-style-type: none">flexibler als hierarchische Datenbankmodelleleistungsfähiger als relationale DatenbankmodelleGute Integrität	<ul style="list-style-type: none">Implementierung, Erweiterung und Wartung aufwändig und kompliziertwird schnell unübersichtlichDatenstruktur bestimmt über Aufbau
Relationales Datenbankmodell	<ul style="list-style-type: none">einfach umzusetzenDaten bleiben weitgehend unabhängig voneinanderSQL-fähig	<ul style="list-style-type: none">weniger leistungsfähig als andere Datenbankmodellekeine Gewährleistung der Datenintegritätfehler- und störungsanfällig
Objektorientiertes Datenbankmodell	<ul style="list-style-type: none">Daten können flexibel repräsentiert werdenunterstützt mehrdimensionale Datenmehrfache Verwendung von Objekten möglich	<ul style="list-style-type: none">Implementierung recht kompliziertgeringe Geschwindigkeit

ID	Anforderungen	Art 1	Abnahmekriterium	Wichtigkeit 2	Dringlichkeit 3
A1	Tabelle Sprache erstellen	F	Tabelle 'sprache' mit folgenden Feldern wird erstellt: <ul style="list-style-type: none">• idlanguage• code• name	5	2
A2	Tabelle Album erstellen	F	Tabelle 'album' mit folgenden Feldern wird erstellt: <ul style="list-style-type: none">• idalbum• title• year• genre• type• cover• initially• inchidance	5	4
A3	Tabelle Tracks erstellen		Tabelle 'tracks' mit folgenden Feldern wird erstellt: <ul style="list-style-type: none">• idtracks• title• artist• length• tracknumber• trackalbum• idlanguage• idgenres	5	4
A4	Tabelle genres erstellen	F	Tabelle 'genres' mit folgenden Feldern wird erstellt: <ul style="list-style-type: none">• name• idgenres	5	3
A5	Untertabelle Album zu genres erstellen	F	Tabelle 'albums_has_genres' mit folgenden Feldern wird erstellt: <ul style="list-style-type: none">• idalbums• idgenres	5	2
A7	Tabelle Personen erstellen	F	Tabelle 'persons' mit folgenden Feldern wird erstellt: <ul style="list-style-type: none">• idpersons• name	5	3
A7	Tabelle Berechtigung erstellen	F	Tabelle 'permissions' mit folgenden Feldern wird erstellt: <ul style="list-style-type: none">• idpermissions	5	3
A8	Tabelle Benutzer erstellen	F	Tabelle 'users' mit folgenden Feldern wird erstellt: <ul style="list-style-type: none">• username• password• permission	5	4
A9	Tabelle Albumtyp erstellen	F	Tabelle 'album_types' mit folgenden Feldern wird erstellt: <ul style="list-style-type: none">• idalbum_types• type	5	2