

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?

CREATE VIEW view\_name AS SELECT ...; – Creates a new view with a specified query.  
 SELECT ... FROM view\_name; – Retrieves data from an existing view.  
 ALTER VIEW view\_name AS SELECT ...; – Modifies an existing view's query.  
 DROP VIEW view\_name; – Deletes an existing view.  
 UPDATE view\_name SET ... WHERE ...; – Updates data in an updatable view.  
 SHOW FULL TABLES WHERE Table\_type = 'VIEW'; – Lists all views in the database.

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**

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

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

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

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 (*kennt*, *liebt*, *hasst*) ausgezeichnet.

```

graph TD
    Alice((Alice)) -- kennt --> Bob((Bob))
    Alice((Alice)) -- kennt --> Carol((Carol))
    Bob((Bob)) -- hasst --> Alice((Alice))
    Bob((Bob)) -- hasst --> Carol((Carol))
    Carol((Carol)) -- liebt --> Alice((Alice))
    Carol((Carol)) -- liebt --> Dave((Dave))
    Dave((Dave)) -- liebt --> Carol((Carol))
  
```

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				

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.

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 schwächen; 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.

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.

ID	Anforderungen	Art 1	Abnahmekriterium	Wichtigkeit 1	Dringlichkeit 2
A1	Tabelle Sprache erstellen	F	Tabelle 'sprache' mit folgenden Feldern wird erstellt: • id • titel • cover • genre	5	2
A2	Tabelle Album erstellen	F	Tabelle 'album' mit folgenden Feldern wird erstellt: • idalbum • title • year • genre • type • cover • initially • inexistence	5	4
A3	Tabelle Tracks erstellen	F	Tabelle 'tracks' mit folgenden Feldern wird erstellt: • idtracks • title • artist • length • tracknumber • albumid • idgenre • idgenre2	5	4
A4	Tabelle genres erstellen	F	Tabelle 'genres' mit folgenden Feldern wird erstellt: • idgenres • name	5	3
A5	Interabelle Album zu genres erstellen	F	Tabelle 'album_has_genres' mit folgenden Feldern wird erstellt: • albumid • genreid • idgenres • idgenres2	5	2
A6	Tabelle Personen erstellen	F	Tabelle 'persons' mit folgenden Feldern wird erstellt: • name • idpersons	5	3
A7	Tabelle Berechtigung erstellen	F	Tabelle 'permissions' mit folgenden Feldern wird erstellt: • name • idpermissions	5	3
A8	Tabelle Benutzer erstellen	F	Tabelle 'users' mit folgenden Feldern wird erstellt: • username • password • permission	5	4
A9	Tabelle Albumtyp erstellen	F	Tabelle 'album_types' mit folgenden Feldern wird erstellt: • idalbum_types • name • type	5	2

Datenbanktyp	Vorteile	Nachteile
Hierarchisches Datenbankmodell	• einfache Struktur • schneller Zugriff • Datenintegrität und Datenunabhängigkeit bleiben erhalten • auch für grosse Datenmengen geeignet	• setzt Kenntnisse der Struktur voraus • jede Beziehung erfordert eigene Definition • pro Satz nur ein Feld und eine Verknüpfung • nachträgliche Strukturänderungen kaum möglich
Netzartiges Datenbankmodell	• flexibler als hierarchische Datenbankmodelle • leistungsfähiger als relationale Datenbankmodelle • Gute Integrität	• Implementierung, Erweiterung und Wartung aufwändig und kompliziert • wird schnell unübersichtlich • Datenstruktur bestimmt über Aufbau
Relationales Datenbankmodell	• einfach umzusetzen • Daten bleiben weitgehend unabhängig voneinander • SQL-fähig	• weniger leistungsfähig als andere Datenbankmodelle • keine Gewährleistung der Datenintegrität • fehler- und störungsanfällig
Objektorientiertes Datenbankmodell	• Daten können flexibel repräsentiert werden • unterstützt mehrdimensionale Daten • mehrfache Verwendung von Objekten möglich	• Implementierung recht kompliziert • geringe Geschwindigkeit