| Type | Action | Command | Description |
|---|---|---|---|
| Login | Login to MySQL CLI | mysql -u root -p | Logs into MySQL using root user and prompts for password. |
| Info | Get current date/time | SELECT NOW(); | Displays current date and time on MySQL server. |
| Info | Show MySQL version | SELECT VERSION(); | Displays installed MySQL version. |
| Info | Show current user | SELECT USER(); | Displays current logged-in user. |
| Database | Create a database | CREATE DATABASE school; | Creates a new database called school. |
| Database | Show all databases | SHOW DATABASES; | Lists all available databases. |
| Database | Use a database | USE school; | Selects school database to work in. |
| Database | Check current database | SELECT DATABASE(); | Shows the active database you're using. |
| Database | Drop a database | DROP DATABASE school; | Deletes the school database permanently. |
| Table | Create table with PRIMARY KEY & NOT NULL | CREATE TABLE students ( id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(100) NOT NULL, age INT NOT NULL ); | Creates students table with id as primary key and required fields. |
| Table | Create table with FOREIGN KEY | CREATE TABLE fees ( fee_id INT AUTO_INCREMENT PRIMARY KEY, student_id INT, amount DECIMAL(10,2), FOREIGN KEY (student_id) REFERENCES students(id) ); | Creates fees table where student_id refers to students.id. |
| Table | Show all tables | SHOW TABLES; | Lists all tables in current database. |
| Table | Show table columns | DESCRIBE students; | Shows column names, types, and constraints. |
| Table | Show table creation SQL | SHOW CREATE TABLE students; | Displays the SQL used to create students table. |
| Table | Rename a table | RENAME TABLE students TO learners; | Changes the name of a table. |
| Table | Drop a table | DROP TABLE students; | Deletes the table and all its data. |
| Table (Alter) | Add column | ALTER TABLE students ADD email VARCHAR(100); | Adds a new column to an existing table. |
| Table (Alter) | Modify column | ALTER TABLE students MODIFY age INT NOT NULL; | Changes the column definition (e.g., make NOT NULL). |
| Table (Alter) | Drop column | ALTER TABLE students DROP COLUMN email; | Removes a column from the table. |
| Table (Alter) | Add PRIMARY KEY | ALTER TABLE students ADD PRIMARY KEY (id); | Adds primary key to column if missing. |
| Table (Alter) | Add FOREIGN KEY | ALTER TABLE fees ADD FOREIGN KEY (student_id) REFERENCES students(id); | Links two tables using foreign key. |
| Constraint | Define NOT NULL | name VARCHAR(100) NOT NULL | Requires value for this column. |
| Constraint | Define PRIMARY KEY | id INT AUTO_INCREMENT PRIMARY KEY | Sets unique identifier and auto-increments. |
| Constraint | Define FOREIGN KEY | FOREIGN KEY (student_id) REFERENCES students(id) | Ensures related records exist in parent table. |
| Constraint | Set DEFAULT value | status VARCHAR(20) NOT NULL DEFAULT 'active' | Assigns default if no value is provided. |
| Row (Data) | Insert new record | INSERT INTO students (name, age) VALUES ('Ali', 16); | Adds a new student entry. |
| Row (Data) | Select all rows | SELECT * FROM students; | Displays all records from table. |

| | | | |
|---|---|---|---|
| Row (Data) | Select specific columns | SELECT name, age FROM students; | Retrieves only specified fields. |
| Row (Data) | Filter with WHERE | SELECT * FROM students WHERE age > 15; | Filters results by a condition. |
| Row (Data) | Update a record | UPDATE students SET age = 17 WHERE name = 'Ali'; | Changes an existing record. |
| Row (Data) | Delete a record | DELETE FROM students WHERE name = 'Ali'; | Removes a specific row. |
| Row (Data) | Delete all rows | TRUNCATE TABLE students; | Deletes all rows and resets auto-increment. |
| Row (Data) | Count rows | SELECT COUNT(*) FROM students; | Returns total number of rows. |
| Aggregate | Sum values | SELECT SUM(amount) FROM fees; | Returns total fee paid. |
| Aggregate | Average value | SELECT AVG(age) FROM students; | Returns average age. |
| Aggregate | Maximum value | SELECT MAX(age) FROM students; | Returns highest age. |
| Aggregate | Minimum value | SELECT MIN(age) FROM students; | Returns lowest age. |
| Group By | Count per group | SELECT age, COUNT(*) FROM students GROUP BY age; | Groups students by age and counts each group. |
| Group By | Sum per group | SELECT student_id, SUM(amount) FROM fees GROUP BY student_id; | Total fees paid by each student. |
| Having | Filter grouped data | SELECT student_id, SUM(amount) AS total FROM fees GROUP BY student_id HAVING total > 2000; | Shows only students who paid more than 2000. |
| Join | INNER JOIN | SELECT s.name, f.amount FROM students s INNER JOIN fees f ON s.id = f.student_id; | Returns students who have matching fee records. |
| Join | LEFT JOIN | SELECT s.name, f.amount FROM students s LEFT JOIN fees f ON s.id = f.student_id; | Returns all students, even if they haven't paid fees (null if no match). |
| Join | RIGHT JOIN | SELECT s.name, f.amount FROM students s RIGHT JOIN fees f ON s.id = f.student_id; | Returns all fee records, even if student is missing (null if no match). |
| Join | FULL OUTER JOIN (simulated) | SELECT s.name, f.amount FROM students s LEFT JOIN fees f ON s.id = f.student_id UNION SELECT s.name, f.amount FROM students s RIGHT JOIN fees f ON s.id = f.student_id; | Combines both left and right join results; includes all students and all fees. |
| Join | JOIN with filter | SELECT s.name, f.amount FROM students s JOIN fees f ON s.id = f.student_id WHERE f.amount > 1000; | Filters joined data using WHERE condition. |
| Subquery | WHERE with subquery | SELECT name FROM students WHERE id IN (SELECT student_id FROM fees); | Gets students who have fee records. |
| Subquery | FROM with subquery | SELECT AVG(age) FROM (SELECT * FROM students WHERE age > 15) AS older_students; | Calculates average age of students older than 15. |
| Subquery | Correlated subquery | SELECT name FROM students s WHERE age > (SELECT AVG(age) FROM students WHERE id <> s.id); | Compares each student's age to average of others. |
| Indexing | Add index to column | CREATE INDEX idx_age ON students(age); | Speeds up queries on the age column. |
| Export | Export database | mysqldump -u root -p school > school.sql | Saves the entire database into a SQL file. |
| Import | Import SQL file | mysql -u root -p school < school.sql | Loads a previously exported SQL file into database. |