[SQL/ MONGODB/ CASSANDRA/ REDIS] Essential Commands

Basic SQL Commands:

- **SELECT**: Retrieve data from a database.
- **INSERT INTO**: Insert data into a table.
- **UPDATE**: Modify existing data in a table.
- **DELETE FROM**: Delete data from a table.
- **CREATE DATABASE**: Create a new database.
- **CREATE TABLE**: Create a new table.
- **ALTER TABLE**: Modify the structure of an existing table.
- **DROP TABLE**: Delete a table.
- **DROP DATABASE**: Delete a database.
- **CREATE INDEX**: Create an index on a table.
- **DROP INDEX**: Delete an index from a table.
- **TRUNCATE TABLE**: Remove all rows from a table.
- **DESCRIBE** or **DESC**: Show table structure.
- **COUNT()**: Count the number of rows in a table.
- **SUM()**: Calculate the sum of values in a column.
- **AVG()**: Calculate the average of values in a column.
- **MAX()**: Find the maximum value in a column.
- **MIN()**: Find the minimum value in a column.
- **GROUP BY**: Group rows based on a column's values.
- **HAVING**: Filter grouped rows.
- **ORDER BY**: Sort rows in the result set.
- **WHERE**: Filter rows based on conditions.
- **JOIN**: Combine data from multiple tables.
- **UNION**: Combine the result sets of multiple SELECT statements.
- **DISTINCT**: Retrieve unique values from a column.
- **LIKE**: Perform pattern matching in a WHERE clause.
- **IN**: Specify multiple values in a WHERE clause.
- **BETWEEN**: Retrieve values within a range.
- **NULL**: Filter for NULL values.
- **CASE**: Perform conditional operations.
- **AS**: Alias columns or tables.

By: Waleed Mousa

- **LIMIT**: Limit the number of rows returned.
- **OFFSET**: Skip a specified number of rows.
- **AS**: Alias columns or tables.
- **LIMIT**: Limit the number of rows returned.
- **OFFSET**: Skip a specified number of rows.

Intermediate SQL Commands:

- **INNER JOIN**: Return only matched rows from multiple tables.
- **LEFT JOIN**: Return all rows from the left table and matched rows from the right table.
- **RIGHT JOIN**: Return all rows from the right table and matched rows from the left table.
- **FULL OUTER JOIN**: Return all rows when there is a match in either the left or right table.
- **SELF JOIN**: Join a table with itself.
- **UNION ALL**: Combine the result sets of multiple SELECT statements, including duplicates.
- **EXISTS**: Check for the existence of rows in a subquery.
- **NOT EXISTS**: Check for the non-existence of rows in a subquery.
- **COUNT(DISTINCT())**: Count distinct values in a column.
- **INNER SELECT**: Create a subquery within a SELECT statement.
- **INSERT INTO SELECT**: Insert data from one table into another based on a SELECT query.
- **UPDATE JOIN**: Update rows in one table based on values from another table.
- **DELETE JOIN**: Delete rows from one table based on values from another table.
- **CASE WHEN**: Perform conditional logic within a query.
- **COALESCE()**: Return the first non-null value in a list.
- **CONCAT()**: Concatenate strings.
- **CAST()**: Convert data types.
- **EXTRACT()**: Extract date or time components.
- **DATE functions**: Perform operations on date values (e.g., DATEADD, DATEDIFF).
- **WINDOW functions**: Perform calculations across a set of table rows.

By: Waleed Mousa

- **TRIGGERS**: Execute actions automatically when specific events occur in the database.

Advanced SQL Commands:

- **INDEX types**: Learn about various index types (e.g., B-tree, Hash).
- **MATERIALIZED VIEWS**: Precomputed views of data stored as tables.
- **TRANSACTIONS**: Manage atomic and consistent database operations.
- **JOINS**: Master complex JOINs, including multiple JOIN clauses.
- **SUBQUERIES**: Create and optimize subqueries.
- **STORED PROCEDURES**: Define and execute server-side procedures.
- **FUNCTIONS**: Create and use custom functions.
- **TRIGGERS**: Write more complex database triggers.
- **VIEWS**: Create and manage views for data abstraction.
- **TEMPORARY TABLES**: Use temporary tables for session-specific data.
- **TRANSACTION ISOLATION**: Understand different isolation levels (e.g., READ COMMITTED, SERIALIZABLE).
- **USER MANAGEMENT**: Manage database users, roles, and permissions.
- **BACKUP and RESTORE**: Learn database backup and restore procedures.
- **OPTIMIZATION**: Optimize SQL queries and database performance.
- **PARTITIONING**: Implement table partitioning for large datasets.
- **ADVANCED AGGREGATE FUNCTIONS**: Learn about additional aggregate functions (e.g., VARIANCE, STDDEV).
- **JSON functions**: Perform operations on JSON data.
- **Regular Expressions (REGEX)**: Use regular expressions in SQL.
- **WITH common_table_expression (CTE)**: Create recursive CTEs.
- **Database Optimization Techniques**: Learn about indexing, query optimization, and database design best practices.

NoSQL Database Commands:

MongoDB Commands:

- **db.createCollection**: Create a new collection.
- **db.collection.insertOne**: Insert a single document into a collection.

By: Waleed Mousa

- **db.collection.insertMany**: Insert multiple documents into a collection.
- **db.collection.find**: Retrieve documents from a collection.
- **db.collection.updateOne**: Update a single document.
- **db.collection.updateMany**: Update multiple documents.
- **db.collection.deleteOne**: Delete a single document.
- **db.collection.deleteMany**: Delete multiple documents.
- **db.collection.aggregate**: Perform aggregation operations.
- **db.collection.createIndex**: Create an index on a collection.
- **db.collection.dropIndex**: Drop an index from a collection.
- **db.collection.distinct**: Find distinct values in a field.
- **db.collection.countDocuments**: Count documents in a collection.
- **db.collection.findOneAndReplace**: Find a document and replace it.
- **db.collection.findOneAndUpdate**: Find a document and update it.
- **db.collection.find().sort()**: Sort query results.
- **db.collection.find().limit()**: Limit the number of results.
- **db.collection.find().skip()**: Skip a specified number of results.
- **db.collection.find().pretty()**: Format query results for readability.
- **db.collection.bulkWrite**: Perform bulk write operations.
- **db.collection.find().explain("executionStats")**: Get query execution statistics.
- **db.collection.find().hint()**: Provide index hint for queries.

Cassandra Commands:

- **CREATE KEYSPACE**: Create a new keyspace.
- **CREATE TABLE**: Create a new table.
- **ALTER TABLE**: Modify table structure.
- **INSERT INTO**: Insert data into a table.
- **UPDATE**: Update data in a table.
- **DELETE**: Delete data from a table.
- **SELECT**: Retrieve data from a table.
- **CREATE INDEX**: Create an index on a table.
- **DROP KEYSPACE**: Delete a keyspace.
- **DROP TABLE**: Delete a table.

By: Waleed Mousa

- **DESCRIBE KEYSPACES**: List keyspaces.
- **DESCRIBE TABLE**: Describe a table.
- **TRUNCATE TABLE**: Remove all data from a table.
- **BATCH**: Perform batch operations.
- **ALLOW FILTERING**: Allow filtering in SELECT queries.

Redis Commands:

- **SET**: Set a key-value pair.
- **GET**: Get the value associated with a key.
- **DEL**: Delete a key.
- **EXPIRE**: Set a key's time to live in seconds.
- **INCR**: Increment the integer value of a key.
- **DECR**: Decrement the integer value of a key.
- **HSET**: Set the field of a hash.
- **HGET**: Get the value of a hash field.
- **HMSET**: Set multiple hash fields.
- **HMGET**: Get multiple hash fields.
- **HGETALL**: Get all fields and values of a hash.
- **RPUSH**: Append a value to a list.
- **LPOP**: Remove and return the first element of a list.
- **ZADD**: Add a member to a sorted set.
- **ZRANGE**: Return a range of members in a sorted set.
- **ZSCORE**: Get the score of a member in a sorted set.
- **SADD**: Add a member to a set.
- **SMEMBERS**: Get all members of a set.
- **SORT**: Sort elements in a list, set, or sorted set.
- **PUBLISH**: Publish a message to a channel.
- **SUBSCRIBE**: Subscribe to a channel.
- **UNSUBSCRIBE**: Unsubscribe from a channel.
- **PSUBSCRIBE**: Subscribe to patterns in channels.
- **PUNSUBSCRIBE**: Unsubscribe from patterns in channels.

Advanced NoSQL Commands:

- **MapReduce (MongoDB)**: Perform MapReduce operations.
- **Cassandra Secondary Indexes**: Create and use secondary indexes.

By: Waleed Mousa

- **Cassandra Batch Statements**: Execute batch operations.
- **Redis Pub/Sub**: Publish/Subscribe to multiple channels.
- **Redis Transactions**: Perform multi-command transactions.
- **Redis Pipeline**: Execute multiple commands in a pipeline.
- **MongoDB Aggregation Pipeline**: Create complex aggregations.
- **Cassandra User-defined Functions (UDFs)**: Create and use UDFs.
- **Cassandra Materialized Views**: Create materialized views.
- **Redis Lua Scripting**: Write and execute Lua scripts.
- **MongoDB Geospatial Queries**: Perform geospatial queries.
- **Redis Streams**: Work with streams and consumer groups.
- **Redis Sentinel**: Monitor and manage Redis high availability.
- **Cassandra TTL (Time To Live)**: Set TTL on data.
- **Cassandra Compaction Strategies**: Configure data compaction.
- **MongoDB Change Streams**: Listen for changes in a collection.
- **MongoDB Full-Text Search**: Perform text-based searches.
- **Cassandra SASI Indexes**: Use secondary indexes for search.
- **Redis Cluster**: Set up a Redis cluster for high availability.
- **Cassandra Materialized Views with TTL**: Combine materialized views with TTL.
- **MongoDB Atlas**: Manage MongoDB in the cloud.
- **Cassandra Repair**: Repair data inconsistencies in Cassandra.
- **Redis Cluster Failover**: Handle failover in a Redis cluster.
- **Cassandra Backups and Restores**: Perform backups and restores.
- **MongoDB Security**: Configure security settings.
- **Redis Security**: Secure Redis instances.
- **Cassandra Authentication and Authorization**: Implement authentication and authorization.

By: Waleed Mousa