ACCESSING DATA

SELECT COMMAND	
Select all rows & columns from a table	`SELECT * FROM Table;`
Select all rows & a specific column from a table	`SELECT Column FROM Table;`
Select all rows, Col1 & Col2 from a table	`SELECT Col1, Col2 FROM Table;`
Using an alias	`SELECT Col AS Alias FROM Table;`
Get all distinct values in a column	`SELECT DISTINCT Col FROM Table;`

WHERE Clause	
Select all rows matching the condition	`SELECT * FROM Table` `WHERE (condition);`

Multiple conditions:
AND: Both conditions must be met
	`SELECT * FROM Table`
	`WHERE (condition1)`
	`AND (condition2);`

Multiple conditions:
Or: Either condition must be met
	`SELECT * FROM Table`
	`WHERE (condition1)`
	`OR (condition2);`

CONDITIONAL OPERATORS	
> Greater than	`WHERE Rating > 10`
< Less than	`WHERE Price < 20.00`
= Equal to (Exact match)	`WHERE Name = "cstutor-sql"`
!= Not equal to	`WHERE Location != "Toronto"`
LIKE OPERATOR	
LIKE: Matching the characters	`WHERE Column LIKE "chars"`
Wildcard for 0 or more of any character Example: Returns 'San Francisco', 'Santa Ana'	`%` `WHERE City LIKE 'San%'`
Wildcard for a single character Example: Returns 3-char string ending with 's'	`_` `WHERE Name LIKE '_ _s'`

IN OR BETWEEN VALUES	
IN: Returns rows where the value in Column matches a value in the list of values	`SELECT * FROM Table WHERE Column IN (list_of_values);`
NOT IN: Returns rows where the value in Column does not match a value in the list of values	`SELECT * FROM Table WHERE Column NOT IN (list_of_values);`
Example: Return rows where the State column matches any value in the list	`SELECT * FROM Table WHERE State IN ('CA', 'NV', 'MN');`
BETWEEN: Values between x and y, upper and lower bound inclusive	`SELECT * FROM Table WHERE Column BETWEEN x AND y;`
NULL VALUES	
IS NULL: Returns rows where the value in the column is NULL	`SELECT * FROM Table WHERE Column IS NULL;`
IS NOT NULL: Returns rows where the value in the column is NOT NULL	`SELECT * FROM Table WHERE Column IS NOT NULL;`

AGGREGATE FUNCTIONS

Syntax	Description	Example
SUM(col)	Adds up all the numeric values in col and returns a single value	`SELECT SUM(TotalCost) FROM Orders;`
AVG(col)	Returns the average value in the specified column	`SELECT AVG(HomePrice)`
MAX(col)	Returns the maximum value in the column	`MAX(Name)`
MIN(col)	Returns the minimum value in the column	`MIN(OrderCount)`
COUNT(col)	Counts the number of non-null rows in the column	`COUNT(EndDate)`

GROUP BY & HAVING

Group By Syntax
(Groups similar rows into summary rows, for aggregate functions)
	`SELECT Column(s)`
	`FROM Table`
	`GROUP BY Column(s)`

Example: Returns number of movies released each year	`SELECT YearReleased, COUNT(*)`
	`FROM Movie`
	`GROUP BY YearReleased;`

Having Syntax
(Similar to WHERE clause but used for aggregate functions)
	`SELECT Column(s)`
	`FROM Table`
	`GROUP BY Column(s)`
	`HAVING condition`

Example: Similar to first example, but limits output to years where more than 2 movies were released	`SELECT YearReleased, COUNT(*)`
	`FROM Movie`
	`GROUP BY YearReleased`
	`HAVING COUNT(*) > 2`

Example: Get all categories where the average price is less than 20	`SELECT Category, AVG(Price)`
	`FROM Dishes`
	`GROUP BY Category`
	`HAVING AVG(Price) < 20`

ORDER BY & LIMIT

Order By Syntax
Default: Ascending
	`SELECT Column(s)`
	`FROM Table`
	`ORDER BY Column [ASC | DESC];`

Example: Order results by Price from low to high	`SELECT * FROM Dish` `ORDER BY Price;`

Order first by Col1, then Col2	`SELECT Column(s)`
	`FROM Table`
	`ORDER BY Col1, Col2;`

Limit syntax	`SELECT Column(s)`
	`FROM Table`
	`LIMIT #ofrows;`

Offset syntax
(Must be used with Limit)
	`SELECT Column(s)`
	`FROM Table`
	`LIMIT #ofrows`
	`OFFSET #ofrows;`

ORDER OF COMMANDS

```
SELECT [DISTINCT] Columns
FROM Tables
[WHERE θ]
[GROUP BY Columns]
[HAVING θ']
[ORDER BY Columns [ASC | DESC]]
[LIMIT # [OFFSET #]]
```

INSERTING, DELETING & UPDATING

Insert data into Table	`INSERT INTO Table(Column1, Column2, ...)` `VALUES (value1, value2, ...);`
Insert values into all columns	`INSERT INTO Table` `VALUES (value1, value2, ...);`
Example:	`INSERT INTO Dish` `VALUES (6, "Falafel", "Dinner", TRUE, 14.99);`
Delete all rows in table	`DELETE FROM Table;`
Delete selected row(s) where condition is met	`DELETE FROM Table` `WHERE condition;`
Example: Deletes 1 dish with matching Id.	`DELETE FROM Dish` `WHERE Id = 6;`
Example: Deletes all dishes with name containing the word salad	`DELETE FROM Dish` `WHERE Name LIKE '%salad%'`

Update Syntax	`UPDATE Table`
	`SET col1 = val1, col2 = val2, ...`
	`[WHERE condition];`

Update Column in every row in Table to *value*	`UPDATE Table` `SET Column = value;`

Update specific row in Table where condition is met	`UPDATE Table`
	`SET Column = value`
	`WHERE condition;`

Example: Updating two columns at once for a row with Id of 15	`UPDATE Dish`
	`SET Name = 'Falafel', Vegetarian = TRUE`
	`WHERE Id = 15;`