SQL Cheat Sheet

Java Concept Of The Day

Introduction

What is SQL?

SQL stands for Structured Query Language. It is a programming language used to store and manipulate the data in relational databases.

SQL Database

CREATE DATABASE : It creates a new SQL database with specified name.

```
CREATE DATABASE database_name;
```

DROP DATABASE : It is used to delete an existing SQL database.

```
DROP DATABASE database_name;
```

BACKUP DATABASE : It is used to create full back up of an existing SQL database.

```
BACKUP DATABASE database_name TO DISK
= 'filepath';
```

BACKUP DATABASE WITH DIFFERENTIAL: It creates differential back up of an existing database. Differential back up backs up only those parts of the database which have been changed since last back up.

```
BACKUP DATABASE database_name TO DISK
= 'filepath' WITH DIFFERENTIAL;
```

SQL Constraints

SQL constraints are used to specify the rules for the columns of a table.

NOT NULL : A column declared with NOT NULL can't have null values.

UNIQUE : A column declared as UNIQUE can't have duplicate values.

DEFAULT : It specifies the default value for a column if no value is provided.

PRIMARY KEY : It declares a column as primary key.

FOREIGN KEY : It declares a column as foreign key.

CHECK : It ensures that values in a column must satisfy the given condition.

SQL Operators

Operators	Symbols
Arithmetic Operators	Add (+), Subtract (-), Multiply (*), Divide (/), Modulus (%)
Bitwise Operators	Bitwise AND (&), Bitwise OR (|), Bitwise exclusive OR (^)
Comparison Operators	Equal To (=), Smaller Than (<), Greater Than (>), Smaller than or equal to (<=), Greater than or equal to (>=), Not equal to (<>)
Compound Operators	Add equals (+=), Subtract equals (-=), Multiply Equals (*=), Divide Equals (/=), Modulus Equals (%=), Bitwise AND equals (&=), Bitwise OR Equals (|*=), Bitwise exclusive OR equals (^-=)
Logical Operators	AND, OR, NOT, ALL, ANY, BETWEEN, IN, EXISTS, LIKE, SOME

SQL Comments

```
--Single Line Comment

/*  Multi
   Line
   Comments  */
```

SQL Views

SQL views are nothing but the virtual tables based on a result set returned by a SQL statement.

CREATE VIEW : It is used to create view.

```
CREATE VIEW view_name AS
SELECT column1, column2 ...
FROM table_name WHERE condition;
```

CREATE OR REPLACE VIEW : This statement is used to update an already existing view.

```
CREATE OR REPLACE VIEW view_name AS
SELECT column1, column2 ...
FROM table_name WHERE condition;
```

DROP VIEW : It is used to remove an already existing view.

```
DROP VIEW view_name;
```

SQL Tables

CREATE TABLE : It is used to create a new table in a database.

```
CREATE TABLE table_name (column1 datatype,
column2 datatype, column3 datatype .... );
```

INSERT INTO : It is used to insert new records into a table.

```
INSERT INTO table_name (column1, column2,
column3 ... ) VALUES (value1, value2, value3
... );
```

DROP TABLE : It is used to delete an existing table from a database.

```
DROP TABLE table_name;
```

TRUNCATE TABLE : It deletes all the data from a table but not the table itself.

```
TRUNCATE TABLE table_name;
```

ALTER TABLE : It is used to add, delete and modify table columns.

```
/* Add a column to a table */
ALTER TABLE table_name
ADD column_name datatype;

/* Delete a column from a table */
ALTER TABLE table_name
DROP COLUMN column_name;

/* Rename a column of a table */
ALTER TABLE table_name
RENAME COLUMN old_name to new_name;

/* Change the datatype of a column */
ALTER TABLE table_name
MODIFY column_name datatype;
```

UPDATE : It is used to modify or update table records.

```
UPDATE table_name
SET column1 = value1, column2 = value2 ...
WHERE condition;
```

DELETE : It is used to delete records from a table.

```
/* Delete all the rows from a table */
DELETE FROM table_name;

/* Delete the rows with condition */
DELETE FROM table_name WHERE condition;
```

SELECT : It is used to retrieve data from a table.

```
/* Select all data from a table */
SELECT * FROM table_name;

/* Select data from specific columns */
SELECT column1, column2 ...
FROM table_name;
```

SELECT DISTINCT : It selects only distinct values from a table.

```
SELECT DISTINCT column1, column2 ...
FROM table_name;
```

SQL Functions

COUNT() : It returns the number of rows which satisfy the given condition.

```
SELECT COUNT(column_name)
FROM table_name WHERE condition;
```

AVG() : It returns average value of a numeric column.

```
SELECT AVG(column_name)
FROM table_name WHERE condition;
```

SUM() : It returns sum of a numeric column.

```
SELECT SUM(column_name)
FROM table_name WHERE condition;
```

MIN() : It returns minimum of a specified column.

```
SELECT MIN(column_name)
FROM table_name WHERE condition;
```

MAX() : It returns maximum of a specified column.

```
SELECT MAX(column_name)
FROM table_name WHERE condition;
```

ROUND() : It is used to round a numeric field.

```
SELECT ROUND(column_name, decimals)
FROM table_name;
```

NOW() : It returns current date and time.

```
SELECT NOW() FROM table_name;
```

SQL Clauses

WHERE : It is used to retrieve or update or delete the records based on some condition. This clause can be used with SELECT, UPDATE and DELETE statements.

```
/* SELECT With WHERE */
SELECT column1, column2 ...
FROM table_name
WHERE condition;

/* UPDATE With WHERE */
UPDATE table_name
SET column1 = value1, column2 = value2 ...
WHERE condition;

/* DELETE With WHERE */
DELETE FROM table_name WHERE condition;
```

ORDER BY : It is used to sort the records in ascending or descending order.

```
SELECT column1, column2 ...
FROM table_name
ORDER BY column1, column2 ... ASC | DESC;
```

GROUP BY : This clause is often used with aggregate functions like SUM(), COUNT(), AVG()... to group the result set by one or two columns.

```
SELECT column_name(s),
aggregate_function_name(column_Name)
FROM table_name
WHERE condition
GROUP BY column_name(s);
```

HAVING : This clause is added to SQL because WHERE can't be used with aggregate functions.

```
SELECT column_name(s),
aggregate_function_name(column_Name)
FROM table_name
WHERE condition
GROUP BY column_name(s)
HAVING condition;
```

SQL Joins

SQL joins are used to combine two or more tables based on a common column between them.

INNER JOIN : It selects the records which are common in both the tables.

```
SELECT column_name(s)
FROM table1 INNER JOIN table2 ON
table1.column_name=table2.column_name;
```

LEFT JOIN : It returns all the records from left table and matching records from right table.

```
SELECT column_name(s)
FROM table1 LEFT JOIN table2 ON
table1.column_name = table2.column_name;
```

RIGHT JOIN : It returns all the records from right table and matching records from left table.

```
SELECT column_name(s)
FROM table1 RIGHT JOIN table2 ON
table1.column_name = table2.column_name;
```

OUTER JOIN or FULL OUTER JOIN : It returns all the records from both the tables.

```
SELECT column_name(s)
FROM table1 FULL OUTER JOIN table2 ON
table1.column_name = table2.column_name;
```

UNION : It is used to combine the result set of two or more select statements.

```
SELECT column_name(s) FROM table1
UNION
SELECT column_name(s) FROM table2;
```

SQL Stored Procedures

SQL stored procedure is a group of pre-compiled SQL statements forming one logical unit and they are stored in a database server and can be called whenever required without compiling again and again.

CREATE PROCEDURE : This statement is used to create stored procedures.

```
CREATE PROCEDURE procedure_name
@parameter_name data_type ....
AS
   BEGIN
     -- SQL statements
   END
```

EXEC : It is used to call stored procedures.

```
EXEC procedure_name;
```

SQL Miscellaneous

AND, OR And NOT : WHERE clause can be used with AND, OR and NOT operators to filter the records with more than one condition.

```
/* AND */
SELECT column1, column2 ...
FROM table_name
WHERE condition1 AND condition2 AND
condition3 ...;

/* OR */
SELECT column1, column2 ...
FROM table_name
WHERE condition1 OR condition2 OR
condition3 ...;

/* NOT */
SELECT column1, column2 ...
FROM table_name
WHERE NOT condition;
```

EXISTS : It is used to test for existence of any records in a sub query.

```
SELECT column_name(s)
FROM table_name
WHERE EXISTS
(SELECT column_name FROM table_name
WHERE condition);
```

AS : It is used to give temporary name called aliases to a table or to a column in a table.

```
/* Alias Column */
SELECT column_name AS alias_name
FROM table_name;

/* Alias Table */
SELECT column_name(s)
FROM table_name AS alias_name;
```

LIKE : It is used with WHERE clause to search for a specified pattern in a column.

```
SELECT column1, column2 ...
FROM table_name
WHERE columnN LIKE pattern;
```

IN : It is used along with WHERE to specify multiple values in WHERE condition.

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1, value2 ...);
```

BETWEEN : It used along with WHERE to filter the values within a specified range.

```
SELECT column_name(s)
FROM table_name
WHERE column_name BETWEEN value1
AND value2;
```

IS NULL And IS NOT NULL : These are used to test for null values.

```
/* IS NULL */
SELECT column_names
FROM table_name
WHERE column_name IS NULL;

/* IS NOT NULL */
SELECT column_names
FROM table_name
WHERE column_name IS NOT NULL;
```

SQL Indexes

SQL indexes are used to speed up search queries in the database tables.

CREATE INDEX : It is used create indexes on the database tables.

```
CREATE INDEX index_name
ON table_name (column1, column2 ...);
```

ALTER INDEX RENAME TO : It is used to rename already existing index.

```
ALTER INDEX old_index_name
RENAME TO new_index_name;
```

DROP INDEX : It is used to remove an already existing index on a table.

```
DROP INDEX Index_Name;
```