Natel Whitaker
SER 401
Task #61

Selection

Select – query data from a single table using SELECT statement.
Order By – sort the result set in ascending or descending order.
Select Distinct – query unique rows from a table using the DISTINCT clause.
Where – filter rows of a result set using various conditions.
AND – filter rows by combining multiple conditions.
OR – combine multiple conditions and filter rows based on at least a specified condition being true.
Limit – constrain the number of rows a query returns and how to get only the necessary data from a table.
Between – test whether a value is in a range of values.
In – check if a value matches any value in a list of values or subquery.
Like – query data based on pattern matching using wildcard characters: percent sign (%) and underscore (_).
Glob – determine whether a string matches a specific UNIX pattern.

Joining Tables

Inner Join – query data from multiple tables using the inner join clause.
Left Join – combine data from multiple tables using the left join clause.
Right Join – combine rows from two tables based on a related column.
Cross Join – show you how to use the cross join clause to produce a cartesian product of result sets of the tables involved in the join.
Self Join – join a table with itself to create a result set that joins rows with other rows within the same table.
Full Outer Join – show you how to use the full outer join in the SQLite.
SQLite joins – recap the joins including inner join, left join, right join, full outer join, and cross join.

Full possible select

```
SELECT DISTINCT column_list
FROM table_list
  JOIN table ON join_condition
WHERE row_filter
ORDER BY column
LIMIT count OFFSET offset
GROUP BY column
HAVING group_filter;
```

Simple select

```
SELECT column_list FROM table;
```

Set Operators

Union – combine result sets of multiple queries into a single result set. We also discuss the differences between UNION and UNION ALL clauses.

Except – compare the result sets of two queries and return distinct rows from the left query that are not output by the right query.

Intersect – compare the result sets of two queries and returns distinct rows that are output by both queries.

Adding data

Insert – insert rows into a table

```
INSERT INTO artists (name)
VALUES('Bud Powell');
```

Update – update existing rows in a table.

```
UPDATE table
SET column_1 = new_value_1,
 column_2 = new_value_2
WHERE
search_condition
ORDER column_or_expression
LIMIT row_count OFFSET offset;
```

Delete – delete rows from a table.

```
DELETE FROM table
WHERE search_condition;
```

Replace – insert a new row or replace the existing row in a table.

```
INSERT OR REPLACE INTO table(column_list)
VALUES(value_list);

REPLACE INTO table(column_list)
VALUES(value_list);
```

Upsert – perform an insert if the row does not exist or update otherwise.

```
INSERT INTO table_name(column_list)
VALUES(value_list)
ON CONFLICT(conflict_column)
DO
    UPDATE SET column_name = expression
    WHERE conflict_condition;
```

RETURNING clause – return the inserted, updated, and deleted rows

```
INSERT INTO table_name(column_list)
VALUES(value_list)
RETURNING *;
```