Cheat Sheet for comprehensive Oracle Database SQL Certified Associate

SQL Basics

SELECT Statement

- **Basic Syntax:**

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

- **Selecting All Columns:**

```
SELECT * FROM table_name;
```

- **Distinct Values:**

```
SELECT DISTINCT column1 FROM table_name;
```

WHERE Clause

- **Basic Syntax:**

```
SELECT column1 FROM table_name WHERE condition;
```

- **Operators:**

- `=` Equal
- `>` Greater than
- `<` Less than
- `>=` Greater than or equal
- `<=` Less than or equal
- `<>` Not equal
- `BETWEEN` Range
- `LIKE` Pattern matching
- `IN` Multiple possible values

ORDER BY Clause

- **Basic Syntax:**

```
SELECT column1 FROM table_name ORDER BY column1 ASC|DESC;
```

- **Multiple Columns:**

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

Data Manipulation Language (DML)

INSERT Statement

- **Basic Syntax:**

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

- **Inserting Multiple Rows:**

```
INSERT INTO table_name (column1, column2) VALUES (value1, value2), (value3, value4);
```

UPDATE Statement

- **Basic Syntax:**

```
UPDATE table_name SET column1 = value1 WHERE condition;
```

- **Updating Multiple Columns:**

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

DELETE Statement

- **Basic Syntax:**

```
DELETE FROM table_name WHERE condition;
```

- **Deleting All Rows:**

```
DELETE FROM table_name;
```

Data Definition Language (DDL)

CREATE TABLE

- **Basic Syntax:**

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

- **Example:**

```
CREATE TABLE employees (
    emp_id NUMBER,
    emp_name VARCHAR2(50),
    hire_date DATE
);
```

ALTER TABLE

- **Add Column:**

```
ALTER TABLE table_name ADD (column_name datatype);
```

- **Modify Column:**

```
ALTER TABLE table_name MODIFY (column_name datatype);
```

- **Drop Column:**

```
ALTER TABLE table_name DROP COLUMN column_name;
```

DROP TABLE

- **Basic Syntax:**

```
DROP TABLE table_name;
```

Data Control Language (DCL)

GRANT Statement

- **Basic Syntax:**

```
GRANT privilege ON object TO user;
```

- **Example:**

```
GRANT SELECT ON employees TO user1;
```

REVOKE Statement

- **Basic Syntax:**

```
REVOKE privilege ON object FROM user;
```

- **Example:**

```
REVOKE SELECT ON employees FROM user1;
```

Data Query Language (DQL)

JOINs

- **INNER JOIN:**

```
SELECT columns FROM table1 INNER JOIN table2 ON table1.column =
table2.column;
```

- **LEFT JOIN:**

```
SELECT columns FROM table1 LEFT JOIN table2 ON table1.column =
table2.column;
```

- **RIGHT JOIN:**

```
SELECT columns FROM table1 RIGHT JOIN table2 ON table1.column =
table2.column;
```

- **FULL OUTER JOIN:**

```
SELECT columns FROM table1 FULL OUTER JOIN table2 ON table1.column = table2.column;
```

Subqueries

- **Basic Syntax:**

```
SELECT column1 FROM table1 WHERE column1 IN (SELECT column2 FROM table2 WHERE condition);
```

- **Correlated Subquery:**

```
SELECT column1 FROM table1 t1 WHERE column1 = (SELECT column2 FROM table2 t2 WHERE t1.column = t2.column);
```

Functions

Aggregate Functions

- **COUNT:**

```
SELECT COUNT(column1) FROM table_name;
```

- **SUM:**

```
SELECT SUM(column1) FROM table_name;
```

- **AVG:**

```
SELECT AVG(column1) FROM table_name;
```

- **MAX:**

```
SELECT MAX(column1) FROM table_name;
```

- **MIN:**

```
SELECT MIN(column1) FROM table_name;
```

String Functions

- **CONCAT:**

```
SELECT CONCAT(column1, column2) FROM table_name;
```

- **SUBSTR:**

```
SELECT SUBSTR(column1, start_position, length) FROM table_name;
```

- **LENGTH:**

```
SELECT LENGTH(column1) FROM table_name;
```

- **UPPER/LOWER:**

```
SELECT UPPER(column1), LOWER(column1) FROM table_name;
```

Date Functions

- **SYSDATE:**

```
SELECT SYSDATE FROM dual;
```

- **ADD_MONTHS:**

```
SELECT ADD_MONTHS(hire_date, 3) FROM employees;
```

- **MONTHS_BETWEEN:**

```
SELECT MONTHS_BETWEEN(SYSDATE, hire_date) FROM employees;
```

- **TO_CHAR:**

```
SELECT TO_CHAR(hire_date, 'YYYY-MM-DD') FROM employees;
```

Transactions

COMMIT

- **Basic Syntax:**

```
COMMIT;
```

ROLLBACK

- **Basic Syntax:**

```
ROLLBACK;
```

SAVEPOINT

- **Basic Syntax:**

```
SAVEPOINT savepoint_name;
```

- **Rollback to Savepoint:**

```
ROLLBACK TO savepoint_name;
```

Indexes

CREATE INDEX

- **Basic Syntax:**

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

DROP INDEX

- **Basic Syntax:**

```
DROP INDEX index_name;
```

Views

CREATE VIEW

- **Basic Syntax:**

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

DROP VIEW

- **Basic Syntax:**

```
DROP VIEW view_name;
```

Sequences

CREATE SEQUENCE

- **Basic Syntax:**

```
CREATE SEQUENCE sequence_name START WITH 1 INCREMENT BY 1;
```

NEXTVAL and CURRVAL

- **NEXTVAL:**

```
SELECT sequence_name.NEXTVAL FROM dual;
```

- **CURRVAL:**

```
SELECT sequence_name.CURRVAL FROM dual;
```

Constraints

PRIMARY KEY

- **Basic Syntax:**

```
CREATE TABLE table_name (
    column1 datatype PRIMARY KEY,
    ...
);
```

FOREIGN KEY

- **Basic Syntax:**

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    FOREIGN KEY (column1) REFERENCES other_table(other_column)
);
```

UNIQUE

- **Basic Syntax:**

```
CREATE TABLE table_name (
    column1 datatype UNIQUE,
    ...
);
```

NOT NULL

- **Basic Syntax:**

```
CREATE TABLE table_name (
    column1 datatype NOT NULL,
    ...
);
```

Miscellaneous

CASE Statement

- **Basic Syntax:**

```
SELECT column1,
       CASE
           WHEN condition1 THEN result1
           WHEN condition2 THEN result2
           ELSE result3
       END AS new_column
FROM table_name;
```

NVL Function

- **Basic Syntax:**

```
SELECT NVL(column1, 'Default_Value') FROM table_name;
```

DECODE Function

- **Basic Syntax:**

```
  SELECT DECODE(column1, value1, result1, value2, result2,
default_result) FROM table_name;
```

Tips and Tricks

- **Use `EXPLAIN PLAN` for query optimization.**
- **Always use `COMMIT` after DML operations to save changes.**
- **Use `ROLLBACK` to undo changes if necessary.**
- **Avoid using `SELECT *` in production queries; specify columns instead.**
- **Use `LIKE` with wildcards (`%` for multiple characters, `_` for single character).**
- **Use `UNION` to combine results of two `SELECT` statements.**
- **Use `GROUP BY` with aggregate functions to summarize data.**
- **Use `HAVING` to filter groups after `GROUP BY`.**
- **Use `EXISTS` to check for the existence of rows in a subquery.**
- **Use `MERGE` to perform `INSERT`, `UPDATE`, or `DELETE` operations based on a condition.**

Examples

Example 1: Basic SELECT with WHERE and ORDER BY

```
SELECT emp_name, hire_date
FROM employees
WHERE hire_date > TO_DATE('2020-01-01', 'YYYY-MM-DD')
ORDER BY hire_date DESC;
```

Example 2: INSERT with Subquery

```
INSERT INTO new_employees (emp_id, emp_name)
SELECT emp_id, emp_name
FROM employees
WHERE department = 'Sales';
```

Example 3: UPDATE with JOIN

```
UPDATE employees e
SET e.salary = e.salary * 1.1
WHERE e.emp_id IN (SELECT m.emp_id FROM managers m WHERE m.department =
'HR');
```

Example 4: DELETE with Subquery

```
DELETE FROM employees
WHERE emp_id IN (SELECT emp_id FROM temp_employees);
```

Example 5: CREATE VIEW with JOIN

```
CREATE VIEW sales_employees AS
SELECT e.emp_id, e.emp_name, s.sales_amount
FROM employees e
JOIN sales s ON e.emp_id = s.emp_id
WHERE e.department = 'Sales';
```

Conclusion

This cheat sheet provides a comprehensive overview of essential SQL commands and concepts for Oracle Database SQL Certified Associate. Use these commands and tips to efficiently manage and query your database.

By Ahmed Baheeg Khorshid

ver 1.0