Agenda

- Indexes
- Constraints
- ~~Alter~~

Indexes

- It is used for faster Searching
- Types of Index
  - Simple Index
  - Unique Index
  - Composite Index
  - Clustered Index

Simple Index

```sql
SELECT * FROM books;
EXPLAIN FORMAT=JSON
SELECT * FROM books WHERE subject = 'C Programming';

CREATE INDEX idx_books_subject ON books(subject);

EXPLAIN FORMAT=JSON
SELECT * FROM books WHERE subject = 'C Programming';

DESCRIBE books;
SHOW INDEXES FROM books;

CREATE INDEX idx_books_author ON books(author DESC);
DESCRIBE books;
SHOW INDEXES FROM books;
```

```sql
EXPLAIN FORMAT=JSON SELECT e.ename, d.dname FROM emps e
INNER JOIN depts d ON e.deptno = d.deptno;

CREATE INDEX idx1 ON emps(deptno);
CREATE INDEX idx2 ON depts(deptno);

EXPLAIN FORMAT=JSON SELECT e.ename, d.dname FROM emps e
INNER JOIN depts d ON e.deptno = d.deptno;
```

Unique Index

- Duplicate values are not allowed inside the column on which unique index is created.

```
CREATE UNIQUE INDEX idx3 ON emps(ename);
DESCRIBE emps;
SELECT * FROM emps;
SELECT * FROM emps WHERE ename = 'Nitin';

INSERT INTO emps VALUES (6, 'Rahul', 70, 1);
-- error: Duplicate entry

INSERT INTO emps VALUES (7, NULL, 50, 5);
-- (multiple) NULL value is allowed, but duplicate is not allowed.

CREATE UNIQUE INDEX idx4 ON emps(mgr);
-- error
```

Composite Index

```
SELECT * FROM emp;
DESCRIBE emp;

EXPLAIN FORMAT=JSON SELECT * FROM emp WHERE deptno = 20 AND job = 'ANALYST';
CREATE INDEX idx_dj ON emp(deptno ASC, job ASC);

DESCRIBE emp;
SHOW INDEXES FROM emp;

EXPLAIN FORMAT=JSON SELECT * FROM emp WHERE deptno = 20 AND job = 'ANALYST';

EXPLAIN FORMAT=JSON SELECT * FROM emp WHERE sal = 5000;

EXPLAIN FORMAT=JSON SELECT * FROM emp WHERE deptno = 20;

EXPLAIN FORMAT=JSON SELECT * FROM emp WHERE job = 'ANALYST';
```

Unique Composite Index

```
CREATE TABLE students(std INT, roll INT, name CHAR(30), marks DECIMAL(5,2));
INSERT INTO students VALUES (1, 1, 's1', 99);
INSERT INTO students VALUES (1, 2, 's2', 96);
INSERT INTO students VALUES (1, 3, 's3', 98);
INSERT INTO students VALUES (2, 1, 's4', 97);
INSERT INTO students VALUES (2, 2, 's5', 95);

CREATE UNIQUE INDEX idx ON students(std,roll);

INSERT INTO students VALUES (1, 2, 's6', 99);
-- error: duplicate combination of std+roll not allowed
```

```
SELECT * FROM students;

INSERT INTO students VALUES (3, 1, 's6', 99);
DESCRIBE students;
```

Clustered Index

- Clustered index is auto-created on Primary key.
- It is internally a unique index that is used to lookup rows quickly on server disk.
- If Primary key is not present in the table, then a hidden (synthetic) column is created by RDBMS and Clustered index is created on it.

Drop Index

```
SHOW INDEXES FROM books;
DROP INDEX idx_books_author ON books;
DESCRIBE books;
```

Constraints

- Their are 5 types of constraints
  - 1. Not Null
  - 2. Unique
  - 3. Primary Key
  - 4. Foreign Key
  - 5. Check
- Based on syntax constraints are classified as
  - 1. Column Level constraints
    - NOT NULL,UNIQUE,PRIMARY, FOREIGN, CHECK

    ```
    CREATE TABLE customers(
        id INT PRIMARY KEY,
        name CHAR(30) NOT NULL,
        email CHAR(40) UNIQUE NOT NULL,
        mobile CHAR(12) UNIQUE,
        addr VARCHAR(100)
    );
    ```

- 2. Table Level constraints
  - UNIQUE,FOREIGN,PRIMARY,CHECK

```
    CREATE TABLE customers(
        id INT,
        name CHAR(30) NOT NULL,
        email CHAR(40) NOT NULL,
        mobile CHAR(12),
        addr VARCHAR(100),
        PRIMARY KEY(id),
        UNIQUE(email),
        UNIQUE(mobile)
);
```

1. NOT NULL

- Null values are not allowed in this column
- can be given at column level only

```
CREATE TABLE temp1(c1 INT, c2 INT, c3 INT NOT NULL);
DESCRIBE temp1;
INSERT INTO temp1 VALUES (1, 1, 1);
INSERT INTO temp1 VALUES (NULL, 2, 2);
INSERT INTO temp1(c1,c3) VALUES (3,3);

SELECT * FROM temp1;

INSERT INTO temp1 VALUES (4, 4, NULL);
-- error: c3 cannot be NULL

INSERT INTO temp1(c1,c2) VALUES (5,5);
-- error: c3 cannot be NULL

SHOW INDEXES FROM temp1;
```

2. Unique

- Cannot have duplicate value in the column.
- However NULL value(s) allowed.
- Unique constraint internally creates unique index.
- Unique constraint on combination of multiple columns internally creates Composite Unique index.
- Must be at table level -- UNIQUE(c1,c2).

```
CREATE TABLE temp2(c1 INT, c2 INT, UNIQUE(c1));
-- CREATE TABLE temp2(c1 INT UNIQUE, c2 INT);

INSERT INTO temp2 VALUES (1, 1);
INSERT INTO temp2 VALUES (2, 2);
INSERT INTO temp2 VALUES (3, 3);
INSERT INTO temp2 VALUES (4, 2);
INSERT INTO temp2 VALUES (3, 5);
-- error: c1 cannot be duplicated

INSERT INTO temp2 VALUES (NULL, 5);
INSERT INTO temp2 VALUES (NULL, 6);

SELECT * FROM temp2;
SHOW INDEXES FROM temp2;

DROP TABLE students;

CREATE TABLE students(std INT,
roll INT,
name CHAR(30),
marks DECIMAL(5,2),
UNIQUE(std,roll)
);

INSERT INTO students VALUES (1, 1, 's1', 99);
INSERT INTO students VALUES (1, 2, 's2', 96);
INSERT INTO students VALUES (1, 3, 's3', 98);
INSERT INTO students VALUES (2, 1, 's4', 97);
INSERT INTO students VALUES (2, 2, 's5', 95);

INSERT INTO students VALUES (1, 2, 's6', 99);
-- error: duplicate combination of std+roll not allowed
```

3. Primary Key

- Primary key is "like" Unique constraint (cannot be duplicated) + NOT NULL constraint (cannot be NULL).
- used to Identity each row/record.
- A table can have a single primary key, but it can have multiple unique key (constraints).
- Primary key can be applied on combination of multiple columns called as composite primary key
- The Primary key internally creates UNIQUE index by name "PRIMARY".

```
-- Natural Primary Key
CREATE TABLE customers(
email CHAR(40) PRIMARY KEY,
password CHAR(40),
name CHAR(40),
addr CHAR(100),
birth DATE
);

-- cannot have multiple primary key but can make multiple UNIQUE + NOTNULL
CREATE TABLE cdac_students(
prn CHAR(16) PRIMARY KEY,
name CHAR(40) NOT NULL,
email CHAR(30) UNIQUE NOT NULL,
mobile CHAR(12) UNIQUE NOT NULL,
```

```sql
addr VARCHAR(100)
);
-- here we can make prn or email or mobile any one of it as primary key.

-- Composite Primary key
DROP TABLE students;
CREATE TABLE students(std INT,
roll INT,
name CHAR(30),
marks DECIMAL(5,2),
PRIMARY KEY(std,roll)
);

INSERT INTO students VALUES (1, 1, 's1', 99);
INSERT INTO students VALUES (1, 2, 's2', 96);
INSERT INTO students VALUES (1, 3, 's3', 98);
INSERT INTO students VALUES (2, 1, 's4', 97);
INSERT INTO students VALUES (2, 2, 's5', 95);

INSERT INTO students VALUES (1, 2, 's6', 99);
-- error: duplicate combination of std+roll not allowed

DESCRIBE students;

SHOW INDEXES FROM cdac_students;
SHOW INDEXES FROM students;
```