Module 5: MySQL Constraints & Indexes

Constraints:

- Constraints are used to enforce the integrity of the data in a table by

- When a constraint is defined at column definition it is called column-level constraint.
- When a constraint is defined at table level using CONSTRAINT keyword it's called table-level constraint.
  - Table-level constraint can use multiple columns
  - Table-level constraint can be provided with some name also.

Unique Constraint:

- Unique constraint

- Unique constraint is either column constraint or table constraint that defines a rule that constrains values in a column or group of columns to be unique.
- If we insert or update a value that causes a duplicate value in unique column, MySQL will issue an error message and reject the change.

- **To define a Unique constraint to a column:**
  - CREATE TABLE table_name(
    - column_name
  -);
- **To define a Unique constraint as table constraint:**
  - CREATE TABLE table_name(
    - ...,
    - col_name data_type,
    - ...,
    - UNIQUE(col_name)
  -);

- If we want to enforce **unique values across columns**, we must define unique constraints as table constraints and separate each column by comma.
  - CREATE TABLE table_name(
  - ...,
  - col_name_3 data_type,
  - col_name_4 data_type,
  - ...,
  - UNIQUE(
  -);

- We can also assign a **name to Unique constraint** using CONSTRAINT clause:
  - CREATE TABLE table_name(
  - …,
  - col_name_1 data_type,
  - col_name_2 data_type,
  - …,
  - **CONSTRAINT constraint_name UNIQUE(col_name_1, col_name_2)**
  -);

- To **see index** we can use:
  - **SHOW INDEX FROM table_name;**

- To **remove index** we can use:
  - DROP INDEX
    OR
  - ALTER TABLE table_name
  - DROP INDEX index_name;

- To **add constraint to existing table**:
  - ALTER TABLE table_name
  - ADD UNIQUE(column_list);

- To **add unique constraint with name**:
  - ALTER TABLE table_name
  - ADD CONSTRAINT constraint_name UNIQUE(column_list);

Primary Key Constraint:

- Primary key constraint requires that each row has a unique value for the column or columns and it does not allow null value.
- Whenever a column is declared as primary following things happen:
  - Column is
  - Column is forced
  - An index is
- Table can have only one primary key.

- To define a **primary key constraint to a column**:
  - CREATE TABLE table_name (
  - …,
  - col_name PRIMARY KEY,
  - …
  -);

- To define a **primary key constraint as table constraint**:
  - CREATE TABLE table_name (
  - …,
  - col_name data_type,
  - …,
  - PRIMARY KEY(col_name)
  -);

- To define a **primary key for multiple columns**:
  - CREATE TABLE table_name (
  - …,
  - col_name_1 data_type
  - col_name_2 data_type,
  - …,
  - PRIMARY KEY(col_name_1, col_name_2)
  -);

- To assign a **name to primary key constraint** we have to use table constraint:
  - CREATE TABLE table_name (
  - …,
  - col_name_1 data_type,
  - col_name_2 data_type,
  - …,
  - CONTRAINT constraint_name PRIMARY KEY(col_name)
  -);

- To **see index** we can use:
  - SHOW INDEX FROM table_name;

- To **drop primary key** :
  - ALTER TABLE table_name
  - DROP

- To **add primary key to existing table**:
  - ALTER TABLE table_name
  - ADD PRIMARY KEY(column_name);

- To **add primary key with some name to existing table**:
  - ALTER TABLE table_name
  - ADD CONSTRAINT constraint_name PRIMARY KEY(column_name);

Foreign Key Constraint:

- Foreign key constraint / Reference Key Constraint requires values in one table match values in another table. This defines the

- To create a foreign key, at the table level we write FOREIGN KEY keywords followed by REFERENCES keyword followed by the name of the related table and the name of the related column in parenthesis.

Actions:

- When a row from the primary table is updated/deleted:
  - If we use **CASCADE** option, then the delete is cascaded to the related rows in the foreign key table
  - If we use **SET NULL** option, then the foreign key column of the foreign key table are set to null.

- To define a **foreign key constraint** :
  - CREATE TABLE table_name (
  - col_name_1 data_type,
  - col_name_2 data_type,
  - …,
  - [CONSTRAINT constraint_name]
  - **FOREIGN KEY (column_name_1 [, column_name_2] …)**
  - **REFERENCES table_name(column_name_1 [, column_name_2]…)**
  - [ON DELETE {CASCADE | SET NULL | **RESTRICT**}]
  - [ON UPDATE {CASCADE | SET NULL | **RESTRICT**}]
  -);

- To **see index** we can use:
  - SHOW INDEX FROM table_name;

- To **drop foreign key** :
  - ALTER TABLE table_name
  - DROP FOREIGN KEY constraint_name;

- To **add foreign key to existing table**:
  - ALTER TABLE table_name
  - ADD FOREIGN KEY (column_list) REFERENCES table_name(column_list)
  - [actions] ;

- To **add foreign key with some name to existing table**:
  - ALTER TABLE table_name
  - ADD CONSTRAINT constraint_name FOREIGN KEY (column_list) REFERENCES table_name(column_list)
  - [actions] ;

Constraint - MCQs

Q1) Which of the following do u need to consider when you make a table in SQL?
Options:

A. Data types
B. Primary keys
C. Default values
D. All of the above.

Solution:

Q2) Which of the following is NOT a type of MySQL constraint?
Options:

A. PRIMARY KEY
B. ALTERNATE KEY
C. FOREIGN KEY
D. UNIQUE

Solution:

Q3) Which are two correct statements about primary key of a table?
Options:

A. Primary keys can contain NULL values
B. Primary keys cannot contain NULL values.
C. A table can have only one primary key with single or multiple fields
D. A table can have multiple primary keys with single or multiple fields

Solution:

Q4) Which two statements are true regarding constraints? (Choose two.)
Options:

A. A table can have only one primary key and one foreign key.
B. A table can have only one primary key but multiple foreign keys.
C. Only the primary key can be defined at the column and table levels.
D. The foreign key and parent table primary key must have the same name.
E. Both primary key and foreign key constraints can be defined at both column and table levels.

Solution:

Index

- Index speeds up searches by providing a way for a DBMS

- MySQL by default creates indexes for Primary Key, Foreign Keys and Unique keys of a table.
- In addition, we can also create indexes for other columns that are frequently used for search conditions.
- Note: avoid creating index on columns that are updated frequently since this slows down insert, update and delete operations.

- To **add index for single column**:
  - CREATE TABLE table_name (
  - column_name_1 data_type,
  - column_name_2 data_type, …,
  - INDEX [INDEX_NAME] (column_name_1)) ;

- To **add index for multiple column**:
  - CREATE TABLE table_name (
  - column_name_1 data_type,
  - column_name_2 data_type, …,
  - INDEX [INDEX_NAME] (column_name_1, column_name_2));
- Keyword KEY may be used instead of INDEX.

- To **find name of index** we can use:
  - SHOW CREATE TABLE table_name\G
  - SHOW INDEX FROM table_name\G

- The **drop the index** we use::
  - **DROP INDEX index_name ON table_name**

OR

  - ALTER TABLE table_name
  - DROP INDEX index_name;

- To **add index**:
  - ALTER TABLE table_name
  - ADD INDEX (column_name);