ADVANCED SQL

TRAINING

- Vasanthi Krishna

Case Study Driven SQL Training covering from refresher of foundation basics and advanced SQL.

Table of Content

- Introduction
- Entity-RelationShip & 3 Normal Forms
- Database & Tables Creation
- Data Definition – INSERT, UPDATE, DELETE
- Data Mainpulation Part1 – SELECT & JOINS
- Data Manipulation Part2 – Subqueries, Derived Tables & SET Operators,
- Q&A

Introduction

- Let's know each other
- Session Plan in Brief
- Structure of the Sessions
  - Case Study Driven
  - Interactive
  - Try it Thyself
- Asks from the learners
  - Raise queries at end of the section or during trials.
  - Plan everyone Time.

Entity-Relation & 3 Normal Forms

- What is an Entity?
- What do we mean by Relation?
- 3 Normal Forms
  - 1 Normal Form :

    A relation is in first normal form if every attribute in that relation is **singled valued attribute**.
  - 2 Normal Form

    A relation is in 2NF if it has **No Partial Dependency,** i.e., no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.
  - 3 Normal Form

    A relation is in third normal form, if there is **no transitive dependency** for non-prime attributes as well as it is in second normal form.

Case Study Time

Let’s START Trying the 3 Normal Forms

Let's understand the sample database

We will be working on a shopping cart database. Do we know any shopping app?

Now let's look at the key tables that the Data Architect is designed for this app.

1. Products – List of Products that is sold through the apps.
2. Product Types – Based on the type of products, additional features might be selected.
3. Customers - Customers registered to the app for buying
4. Orders - Orders placed by Customers
5. Delivery Invoice – Invoice for the products delivered.
6. Warehouse – Place where the products are stocked.

The database design is simplified version of a real-life complex data architecture and primarily aims to aid in clearing the advance SQL Data manipulation and data programming concepts.

Let's Create our Database

- Databases are generally created by Database Administrator to ensure
  - appropriate memory, storage and access
  - Adequate up-time
  - Backup and Restoration
- To create a database, syntax as below:

```
CREATE DATABASE [IF NOT EXISTS] database_name
[CHARACTER SET charset_name]
[COLLATE collation_name]
```

- In MySQL, creation of database is synonymous to create a schema.
- Our Database is . Let's create using the MySQL workbench.

Core Data Objects - Tables

- Once database is create, we are required to create the data objects. In RDMS, the core is tables. All data are stored in tables as rows (data) and columns (attributes)
- To create a table, indicates we need to know
  - what that table is meant to store,
  - the attributes for that table,
  - data type for the attributes and then
  - the keys for the table.
- Frequent Actions that happens on Tables are
  - CREATE TABLE – To create new tables.
  - ALTER TABLE – helps to change the structure of a table.
  - RENAME TABLE – helps to rename a table.
  - DROP TABLE – show you how to remove existing tables using DROP TABLE statement.
  - TRUNCATE TABLE – show you how to delete all data from a table fast and more efficient using the TRUNCATE TABLE statement.
- Frequent Actions on columns of Table are
  - ALTER TABLE ADD Column – helps to add one or more columns to an existing table.
  - Generated columns – helps to use generated columns to store data computed from an expression or other columns..
  - ALTER TABLE DROP COLUMN – helps to remove one or more columns from a table.

CREATE TABLE

- To create a table, indicates we need to know
  - what that table is meant to store,
  - the attributes for that table,
  - data type for the attributes and then
  - the keys for the table.
- Let's create a table

```
CREATE TABLE [IF NOT EXISTS] table_name(
   column_1_definition,
   column_2_definition,
   ...,
   table_constraints
)
```

CREATE TABLE - Contd

- We are required to create columns with multiple attributes to each column
- column_name data_type(length) [NOT NULL] [DEFAULT value] [AUTO_INCREMENT] column_constraint;
  - Column name : Name of the column
  - data_type(length) : for the column, type of data like char, varchar(), int, float,date as well as the length
  - NOT NULL : This indicates that this column needs to have a value.
  - DEFAULT : This is the default value the column will have in case not entered.
  - AUTO_INCREMENT : indicates that the value of the column is incremented by one automatically whenever a new row is inserted into the table. Each table has a maximum one AUTO_INCREMENT column.
  - Column_constraint: These are constraints on the particular column
- table Constraints can be applied to multiple columns like Primary key, Unique, Foreign key
  - PRIMARY KEY (col1, col2) : Combination of both col1 and col2 will be used as primary key
  - UNIQUE (col1, col2) : combination of col1 and col2 together will be checked for uniqueness on each new entry

Let's create our tables in Myshop db.

- Find the SQL Script file "CreateSampleDB-Tables".
- Open the Script file in a Note pad.
- Copy paste all the create table commands to MySQL Workbench.

IS THERE AN ALTERNATIVE?

Data Creation

- We have the tables ready. How can we explore it further? We need some data.
- Data Creation is facilitated by the below concepts
- INSERT – use various forms of the INSERT statement to insert data into a table.
  - INSERT Multiple Rows – insert multiple rows into a table.
  - INSERT INTO SELECT – insert data into a table from the result set of a query.
  - INSERT IGNORE – explain you the INSERT IGNORE statement that inserts rows into a table and ignores rows that cause errors.
- UPDATE – learn how to use UPDATE statement and its options to update data in database tables.
  - UPDATE JOIN – show you how to perform cross-table update using UPDATE JOIN statement with INNER JOIN and LEFT JOIN.
- DELETE – show you how to use the DELETE statement to delete rows from one or more tables.
  - ON DELETE CASCADE – learn how to use ON DELETE CASCADE referential action for a foreign key to delete data from a child table automatically when you delete data from a parent table.
  - DELETE JOIN – show you how to delete data from multiple tables.
  - REPLACE – learn how to insert or update data depends on whether data exists in the table or not.
  - Prepared Statement – show you how to use the prepared statement to execute a query.

INSERT INTO

- INSERT – Different forms of INSERT statement to insert data into a table.

```
INSERT INTO table(c1,c2,...)
VALUES (v1,v2,...);
```

- INSERT Multiple Rows – insert multiple rows into a table.

```
INSERT INTO table(c1,c2,...)
VALUES
  (v11,v12,...),
  (v21,v22,...),
   ...
  (vnn,vn2,...);
```

- INSERT INTO SELECT – insert data into a table from the result set of a query.
- INSERT IGNORE – explain you the INSERT IGNORE statement that inserts rows into a table and ignores rows that cause errors.

UPDATE

- UPDATE – Basic UPDATE statement and its options to update data in database tables.

```
UPDATE [LOW_PRIORITY] [IGNORE] table_name
SET
   column_name1 = expr1,
   column_name2 = expr2,
   ...
[WHERE
   condition];
```

There are two modifiers that helps more

LOW_PRIORITY – This delays the update delay the update until there is no connection reading data from the table.

IGNORE - UPDATE statement to continue updating rows even if errors occurred. The rows that cause errors such as duplicate-key conflicts are not updated

DELETE

- DELETE – use the DELETE statement to delete rows from one or more tables.

```
DELETE FROM table_name
WHERE condition;
LIMIT number of rows
```

- ON DELETE CASCADE - use ON DELETE CASCADE referential action for a foreign key to delete data from a child table automatically when you delete data from a parent table. When you define your child table with create table, there you need to mention about Cascade on delete.
- DELETE JOIN – show you how to delete data from multiple tables

```
DELETE T1, T2
FROM T1
INNER JOIN T2 ON T1.key = T2.key
WHERE condition;

DELETE T1
FROM T1
      LEFT JOIN
   T2 ON T1.key = T2.key
WHERE
   T2.key IS NULL;
```

Case Study Time

Let’s TRY OURSELVES

Data Manipulation – Part1

- Data Manipulation is all about to extract data from the database for varying needs.
- Extraction of data is all about below sections

Section 1. Querying data

SELECT FROM – Use SELECT FROM statement to query the data from a single table.

SELECT – SELECT statement without referencing a table.

Section 2. Sorting data

ORDER BY – sort the output result set using ORDER BY clause..

Data Manipulation – Part1

Section 3. Filtering data

WHERE – WHERE clause helps to filter rows based on specified conditions.

SELECT DISTINCT –the DISTINCT operator in the SELECT statement to eliminate duplicate rows in a result set.

AND – AND operator to combine Boolean expressions to form a complex condition.

OR – OR operator. combine the OR operator with the AND operator to filter data.

IN – IN operator in the WHERE clause to determine a value matches any value in a set.

NOT IN – negate the IN operator using the NOT operator to check if a value doesn't match any value in a set.

BETWEEN – query data based on a range using BETWEEN operator.

LIKE – to query data based on a pattern.

LIMIT – to constrain the number of rows returned by SELECT statement

IS NULL – check if a value is NULL or not by using IS NULL operator.

Data Manipulation – Part1

Section 4. Joining tables

Table & Column Aliases – create table and column aliases.

Joins – Understand the different joins to extract data from multiple tables using different conditions.

INNER JOIN – query rows from table that has matching rows in another table.

LEFT JOIN – all rows from the left table and matching rows from the right table or null if no matching rows found in the right table.

RIGHT JOIN – return all rows from the right table and matching rows from the left table or null if no matching rows found in the left table.

CROSS JOIN – make a Cartesian product of rows from multiple tables.

Self-join – join a table to itself using table alias and connect rows within the same table using inner join and left join.

Section 5. Grouping data

GROUP BY – how to group rows into groups based on columns or expressions.

HAVING – filter the groups by a specific condition.

ROLLUP – generate multiple grouping sets considering a hierarchy between columns specified in the GROUP BY clause.

Select Statement

```
SELECT [ALL | DISTINCT | DISTINCTROW ]
select_expr [, select_expr] ... [into_option]
[FROM table_references]
[WHERE where_condition
[AND]/[OR]/[BETWEEN]/[ISNULL]/[IN]/[NOT IN]/]
[GROUP BY {col_name | expr | position}, ...
[WITH ROLLUP]]
[HAVING where_condition]
[ORDER BY {col_name | expr | position} [ASC | DESC], ... [WITH ROLLUP]]
[LIMIT {[offset,] row_count | row_count OFFSET offset}] [into_option]
```

Case Study Time

Let’s TRY OURSELVES

Data Manipulation – Part2

Section 6. Subqueries

Subquery – Nest a query (inner query) within another query (outer query) and use the result of the inner query for the outer query.

Derived table – create derived table concept and use it to simplify complex queries.

EXISTS – test for the existence of rows.

Section 7. Set operators

UNION and UNION ALL – combine two or more result sets of multiple queries into a single result set.

INTERSECT – to simulate the INTERSECT operator.

MINUS – explain the SQL MINUS operator indicating of how to ignore some data from result set.

Case Study Time

Let’s TRY OURSELVES

Q&A