Indexes and Triggers

Indexes

- An Index is a structure that provides accelerated access to the rows (records) of a table based on the values of one or more columns.

- The presence of an index can significantly improve the performance of a query.

- However, as indexes may be updated by the system every time the underlying tables are updated, additional overheads may be incurred.

Indexes

- MySQL creates indexes automatically for all primary key columns and will create indexes for foreign key columns if the underlying table structure is innoDB.

- Indexes are usually created to satisfy particular search criteria after the table has been in use for some time and has grown in size.

Index syntax

```
CREATE [UNIQUE] INDEX indexName ON
TableName (columnName [ASC|DESC] [....])
```

- Indexes can be created on base tables only (and not Views).
- If the UNIQUE clause is used, uniqueness of the indexed column or combination of columns will be enforced by the DBMS. This is required for the primary key (and alternate key) columns (fields).

Index syntax

- To create an index on the book table for the title to speed up searching:

```
create index booktitleind on book(title);
```

Removing an Index

- If we create an index for a base table and later decide that it is no longer needed, we can use the DROP INDEX command combined with the ALTER TABLE statement to remove the index from the database.

```
ALTER TABLE TableName DROP INDEX indexName
```

- The following statement would remove the index created in the previous example:

```
alter table book drop index booktitleind;
```

Triggers

- A SQL trigger is a set of SQL statements stored in the database catalog.
- A SQL trigger is executed or fired whenever an event associated with a table occurs e.g., INSERT, UPDATE or DELETE.
- A trigger can be defined to be invoked either before or after the data is changed.

MySQL trigger syntax

- In order to create a new trigger, you use the CREATE TRIGGER statement. The following illustrates the syntax of the CREATE TRIGGER statement:

```
CREATE TRIGGER
  trigger_name trigger_time trigger_event
ON table_name
FOR EACH ROW
BEGIN
...
END;
```

MySQL trigger example

- We will create a trigger in MySQL to log the changes of the Book table in the library database.

- To do this we will create a new table, **book_audit**:

```
CREATE TABLE book_audit (
    id INT AUTO_INCREMENT PRIMARY KEY,
    ISBN varchar(15) not null,
    price decimal(5,2),
    changedate DATETIME DEFAULT NULL,
    action VARCHAR(50) DEFAULT NULL
);
```

MySQL trigger example

- Now, we will create a trigger **before_book_update**:

```
DELIMITER $$
CREATE TRIGGER before_book_update
    BEFORE UPDATE ON book
    FOR EACH ROW
BEGIN
    INSERT INTO book_audit
    SET action = 'update',
        ISBN = OLD.ISBN,
    price = OLD.price,
        changedate = NOW();
END$$
DELIMITER ;
```

Removing Triggers

- To remove a trigger, use the DROP TRIGGER statement as follows:

```
drop trigger before_book_update;
```

Triggers and security

- In relation to database security, triggers can be useful e.g. to capture details on what changes were made and by whom