<epam>

Physical Modelling Sample

Relational Databases Basics

TRAINING
CENTER

<epam>

Disclaimer (yes, its similar to the one you've seen recently)

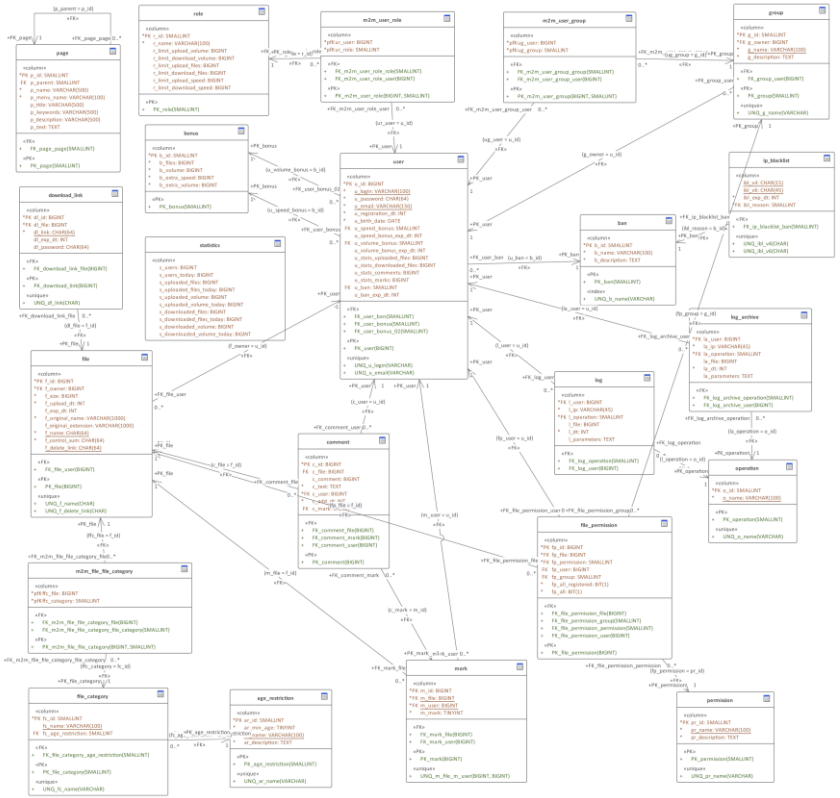
In real life this is a LONG iterative process. It may take days and months. And even here (with this extremely simplified sample) you'll have to spend a lot of time in order to comprehend all the information.

Just a quick reminder on the initial setup

We are working on the database for the “File Exchange” service. Here is the info from the customer:

1. The application may contain several pages (the quantity, the hierarchy, and the contents may vary).
2. Application users may create groups and join such groups.
3. Each user may have several roles with a set of permissions for each role.
4. Uses may upload and download files, share files with specific users, groups of users, and the whole world.
5. Users may comment files.
6. Each file has a rating.
7. There may be replies to comments (and other replies) – up to 10 levels of nesting depth.
8. Each file must belong to a category, which determine the set of permissions and limitations.
9. The application shall log all actions of all users.
10. There must be possibility to ban users, groups of users, and non-registered users (by ip address).
11. The application shall display (with minimum time delay) the following statistics: total users, total uploaded files quantity and volume, total downloaded files quantity and volume.

And that was the result of datalogical modelling



So, here we have to set…

- Access permissions
- Encodings
- Storage engines
- Indexes
- DB/DBMS settings

Access permissions

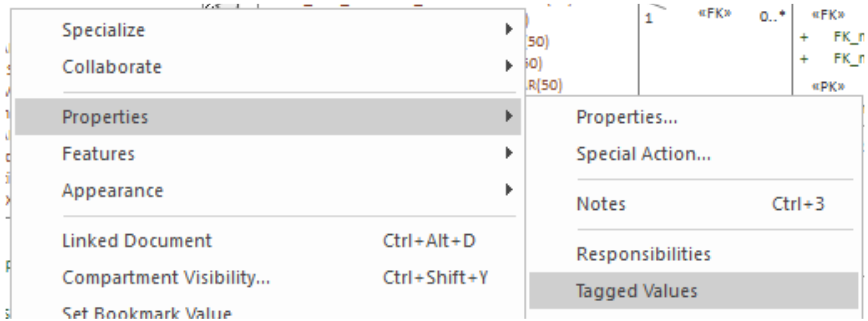
	Application	Guest	Registered user	Moderator	Administrator
age_restriction	R	R	R	CRUD	CRUD
ban	R	R	R	R	CRUD
bonus	R	R	R	R	CRUD
comment	R	R	R	CRUD	CRUD
download_link	R	R	R	CRUD	CRUD
file	R	R	CRUD	CRUD	CRUD
file_category	R	R	R	CRUD	CRUD
file_permission	R	R	CRUD	CRUD	CRUD
group	R	R	R	CRUD	CRUD
ip_blacklist	CRUD	-	-	-	CRUD
log	C	-	-	-	CRUD
log_archive	C	-	-	-	CRUD
m2m_file_file_category	R	R	CRUD	CRUD	CRUD
m2m_user_group	R	R	R	CRUD	CRUD
m2m_user_role	R	R	R	CRUD	CRUD
mark	R	R	CRUD	CRUD	CRUD
operation	R	-	-	-	CRUD
page	R	R	R	CRUD	CRUD
permission	R	R	R	R	CRUD
role	R	R	R	R	CRUD
statistics	R	R	R	R	CRUD
user	CRUD	R	CRUD	CRUD	CRUD

Access permissions

```
-- 1) "Application" role permissions:
DROP USER IF EXISTS 'feapp'@'localhost';
CREATE USER 'feapp'@'localhost' IDENTIFIED BY '<complex password>';

GRANT SELECT ON `age_restriction` TO 'feapp'@'localhost';
GRANT SELECT ON `ban` TO 'feapp'@'localhost';
GRANT SELECT ON `bonus` TO 'feapp'@'localhost';
GRANT SELECT ON `comment` TO 'feapp'@'localhost';
GRANT SELECT ON `download_link` TO 'feapp'@'localhost';
GRANT SELECT ON `file` TO 'feapp'@'localhost';
GRANT SELECT ON `file_category` TO 'feapp'@'localhost';
GRANT SELECT ON `file_permission` TO 'feapp'@'localhost';
GRANT SELECT ON `group` TO 'feapp'@'localhost';
GRANT INSERT, SELECT, UPDATE, DELETE ON `ip_blacklist`
                                     TO 'feapp'@'localhost';
GRANT INSERT ON `log` TO 'feapp'@'localhost';
GRANT INSERT ON `log_archive` TO 'feapp'@'localhost';
GRANT SELECT ON `m2m_file_file_category` TO 'feapp'@'localhost';
GRANT SELECT ON `m2m_user_group` TO 'feapp'@'localhost';
GRANT SELECT ON `m2m_user_role` TO 'feapp'@'localhost';
GRANT SELECT ON `mark` TO 'feapp'@'localhost';
GRANT SELECT ON `operation` TO 'feapp'@'localhost';
GRANT SELECT ON `page` TO 'feapp'@'localhost';
GRANT SELECT ON `permission` TO 'feapp'@'localhost';
GRANT SELECT ON `role` TO 'feapp'@'localhost';
GRANT SELECT ON `statistics` TO 'feapp'@'localhost';
GRANT INSERT, SELECT, UPDATE, DELETE ON `user` TO 'feapp'@'localhost';
GRANT CREATE USER ON *.* TO 'feapp'@'localhost' WITH GRANT OPTION;

-- 2) "Guest", "Registered user", and "Moderator" roles permissions setup
--    looks similar.

-- 3) "Administrator" role permissions:
DROP USER IF EXISTS 'feadmin'@'localhost';
CREATE USER 'feadmin'@'localhost' IDENTIFIED BY '<complex password>';

GRANT ALL PRIVILEGES ON * TO 'feadmin'@'localhost';
```

Encodings (using Sparx Enterprise Architect)

Step 1



Step 2 (repeat for all encodings settings, there may be many)

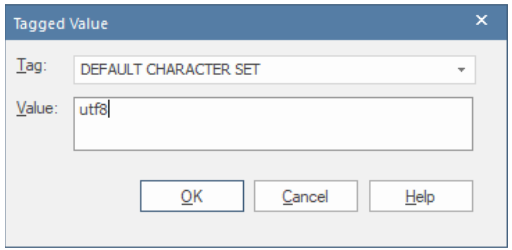
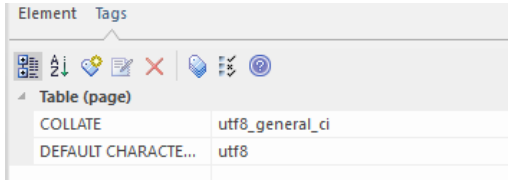


Table	
CHARSET	utf8
COLLATE	utf8_general_ci
ENGINE	InnoDB

Result



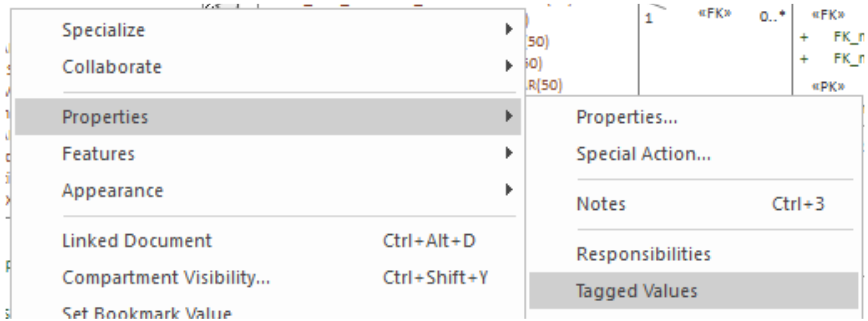
Sparx EA 15,
MySQL 8

Encodings (using stored procedure)

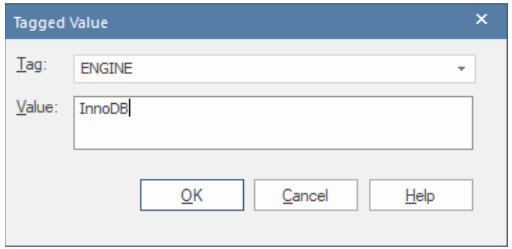
```
DELIMITER $$
CREATE PROCEDURE SET_ENCODING_TO_ALL_TABLES
   (IN default_charset_name VARCHAR(150), IN collation_name VARCHAR(150))
BEGIN
 DECLARE done INT DEFAULT 0;
 DECLARE tbl_name VARCHAR(200) DEFAULT '';
 DECLARE all_tables_cursor CURSOR FOR
  SELECT `table_name`
    FROM `information_schema`.`tables`
    WHERE `table_schema` = DATABASE()
      AND `table_type` = 'BASE TABLE';
 DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

 OPEN all_tables_cursor;
 tables_loop: LOOP
  FETCH all_tables_cursor INTO tbl_name;
  IF done
   THEN LEAVE tables_loop;
  END IF;

  SET @alter_table_query = CONCAT('ALTER TABLE `', tbl_name,
      '` CONVERT TO CHARACTER SET \'', default_charset_name,
      '\' COLLATE \'', collation_name, '\'');

  PREPARE alter_table_stmt FROM @alter_table_query;
  EXECUTE alter_table_stmt;
  DEALLOCATE PREPARE alter_table_stmt;
 END LOOP tables_loop;
 CLOSE all_tables_cursor;
END;
$$
DELIMITER ;

CALL SET_ENCODING_TO_ALL_TABLES('utf8', 'utf8_general_ci');
```

Storage engines (using Sparx Enterprise Architect)

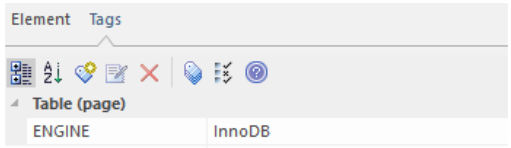
Step 1



Step 2



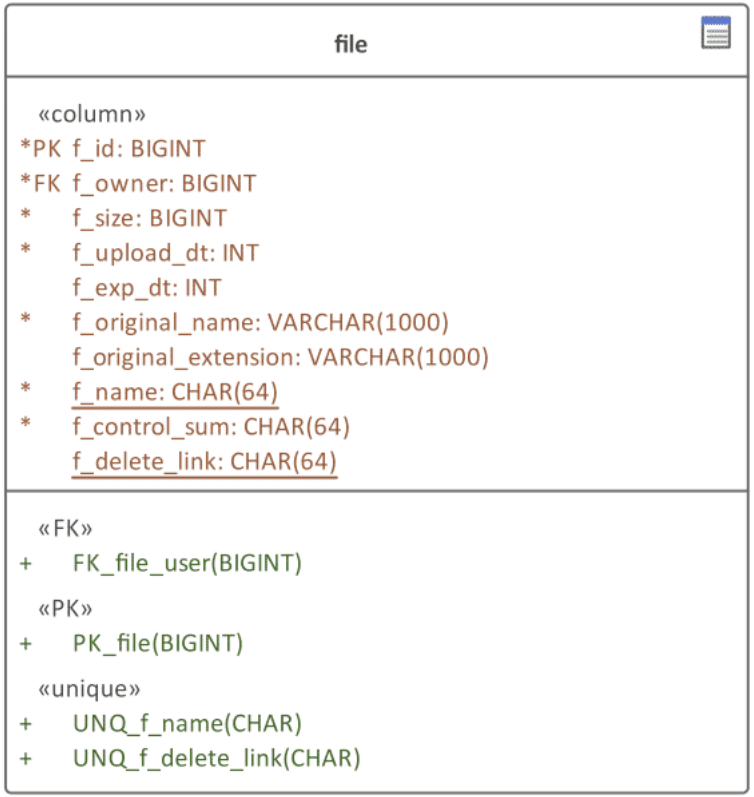
Result



Storage engines (and encodings) (using stored procedure)

```
DELIMITER $$
CREATE PROCEDURE SET_ENCODING_AND_STORAGE_ENGINE_TO_ALL_TABLES
          (IN default_charset_name VARCHAR(150),
           IN collation_name VARCHAR(150),
           IN storage_engine VARCHAR(150))
BEGIN
 DECLARE done INT DEFAULT 0;
 DECLARE tbl_name VARCHAR(200) DEFAULT '';
 DECLARE all_tables_cursor CURSOR FOR
  SELECT `table_name`
    FROM `information_schema`.`tables`
    WHERE `table_schema` = DATABASE()
      AND `table_type` = 'BASE TABLE';
 DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

 OPEN all_tables_cursor;
 tables_loop: LOOP
  FETCH all_tables_cursor INTO tbl_name;
  IF done
   THEN LEAVE tables_loop;
  END IF;

  SET @alter_table_encoding_query = CONCAT('ALTER TABLE `', tbl_name,
    '` CONVERT TO CHARACTER SET \'', default_charset_name,
    '\' COLLATE \'', collation_name, '\'');
  SET @alter_table_engine_query = CONCAT('ALTER TABLE `', tbl_name,
    '` ENGINE = \'', storage_engine, '\'');

  PREPARE alter_table_encoding_stmt FROM @alter_table_encoding_query;
  PREPARE alter_table_engine_stmt FROM @alter_table_engine_query;

  EXECUTE alter_table_encoding_stmt;
  EXECUTE alter_table_engine_stmt;

  DEALLOCATE PREPARE alter_table_encoding_stmt;
  DEALLOCATE PREPARE alter_table_engine_stmt;
 END LOOP tables_loop;
 CLOSE all_tables_cursor;
END;
$$
DELIMITER ;

CALL SET_ENCODING_AND_STORAGE_ENGINE_TO_ALL_TABLES('utf8',
                                        'utf8_general_ci', 'InnoDB');
```

Indexes



This is `file` table current state. And we have to speed up the following queries:

- expired files deletion;
- file search by name and/or extension;
- file ordering by size;
- file ordering by creation datetime.

Indexes

Let's make at least one experiment…

```
EXPLAIN DELETE FROM `file` WHERE `f_exp_dt` <= UNIX_TIMESTAMP()
```

select_type	table	type	possible_keys	key	key_len	ref	rows	filtered	Extra
DELETE	file	ALL	NULL	NULL	NULL	NULL	1000000	100.00	Using where

Without `IDX_f_exp_dt` index

With `IDX_f_exp_dt` index

W/o index, s	With index. S	Faster
7.00951E-03	5.79357E-05	120.98775021274

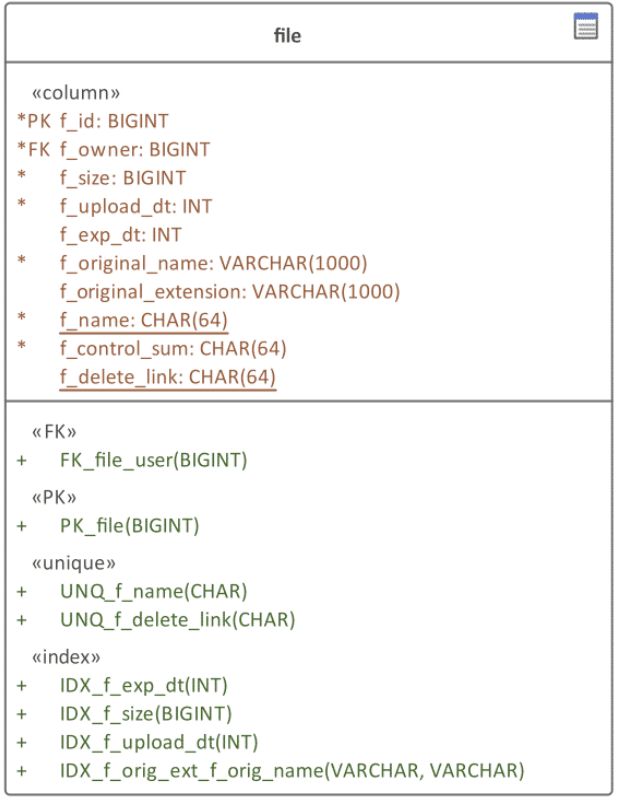
select_type	table	type	possible_keys	key	key_len	ref	rows	filtered	Extra
DELETE	file	ALL	IDX_f_exp_dt	IDX_f_exp_dt	4	NULL	1000000	50.00	Using where

Indexes

Indexes creation code

```
ALTER TABLE `file`
 ADD INDEX `IDX_f_exp_dt` (`f_exp_dt` ASC);

ALTER TABLE `file`
 ADD INDEX `IDX_f_size` (`f_size` ASC);

ALTER TABLE `file`
 ADD INDEX `IDX_f_upload_dt` (`f_upload_dt` ASC);

ALTER TABLE `file`
 ADD INDEX `IDX_f_orig_ext_f_orig_name`
         (`f_original_extension` ASC,`f_original_name` ASC);
```

New table state



DB/DBMS settings

```
-- Approach 1:
SET character_set_server = utf8mb4
SET collation_server = utf8mb4_general_ci

-- Approach 2:
SET NAMES utf8mb4 COLLATE utf8mb4_general_ci;

-- Approach 3 (permanent settings change):
SET PERSIST character_set_server = utf8mb4
SET PERSIST collation_server = utf8mb4_general_ci
```

```
In my.ini under [mysqld] section add the following options:
character_set_server with utf8mb4 value,
collation_server with utf8mb4_general_ci value.
```

NEVER give a piece of ready-to-use config to avoid “copy-paste without thinking”!

Quick live demo…

<epam>

Physical Modelling Sample

Relational Databases Basics

TRAINING
CENTER

<epam>