<epam>

Normalization Requirements

Relational Databases Basics

TRAINING
CENTER

<epam>

Before we begin…

It is extremely useful to remember that during normalization process we can achieve both important goals:

- eliminate data operation anomalies;
- significantly improve the whole database quality.

We shall discuss it now.

Disclaimer

In the next several minutes we shall review a lot of so called “normalization requirements”. It’s hard to comply with them all in a single database, but this is not the goal.

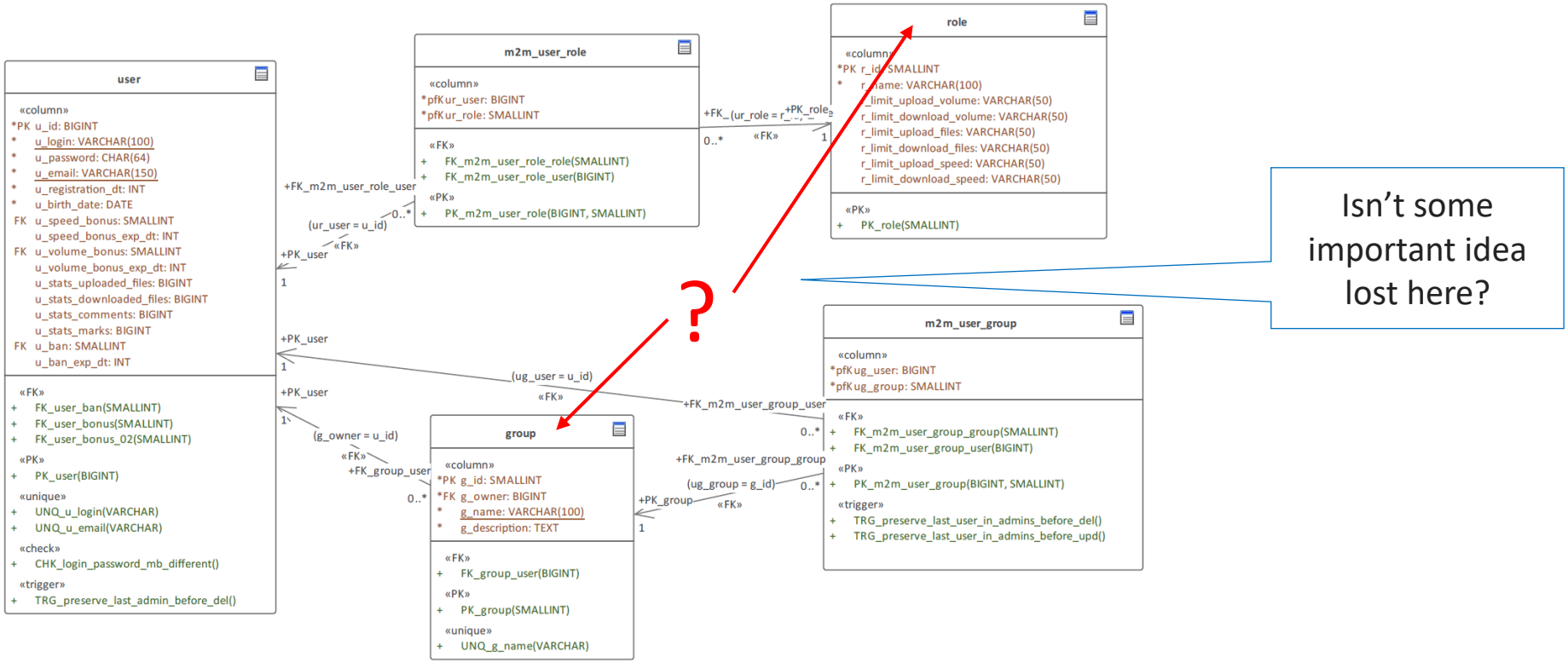
The idea is to determine our main quality goals and follow those “normalization requirements” that bring us closer to that goals.

Read and remember!

Normalization – a process of decomposition of relation variable R into projections R_1, R2, ..., R_n, such that:

- the join of R_1, R_2, ..., R_n is guaranteed to be equal to R;
- each of R_1, R_2, ..., R_n is needed in order to provide that guarantee;
- at least one of R_1, R_2, ..., R_n is at a higher level of normalization than R is.

Subject matter constraints preservation



Fundamental database requirements compliance

1. Subject matter adequacy
2. Technical usability
3. Performance
4. Data safety

Primary keys minimality

Does a table need a Primary Key at all?

Is this Primary Key small enough?

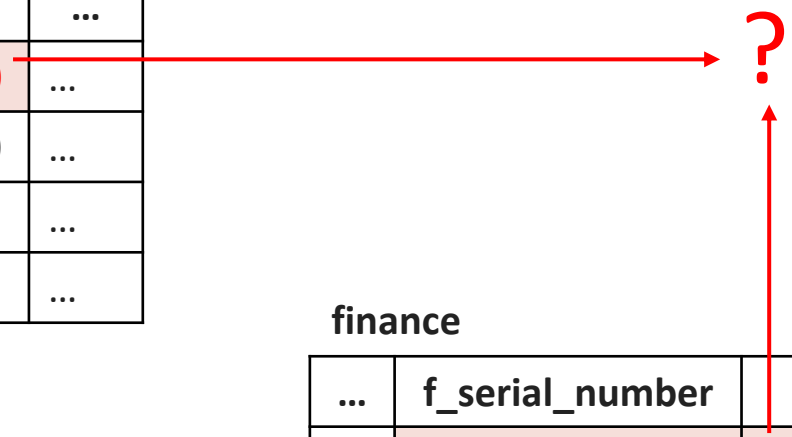
Is this Primary Key big enough?

Can we preserve data range making the Primary Key smaller?

Data storage non-redundancy

contract

…	c_serial_number	c_sum	…
…	AC345347856DF	34 000 000	…
…	DF345345652YH	12 000 000	…
…	AA345235235KL	32 511 012	…
…	GT456345342UT	41 356 343	…

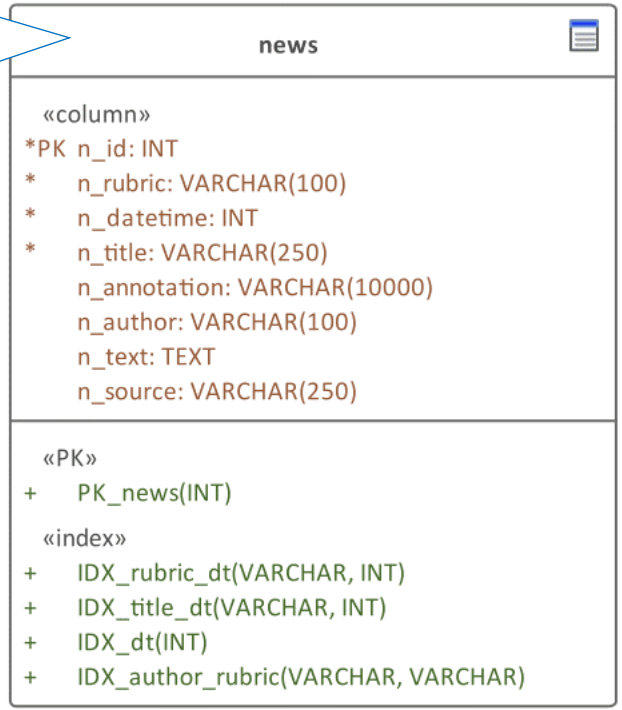


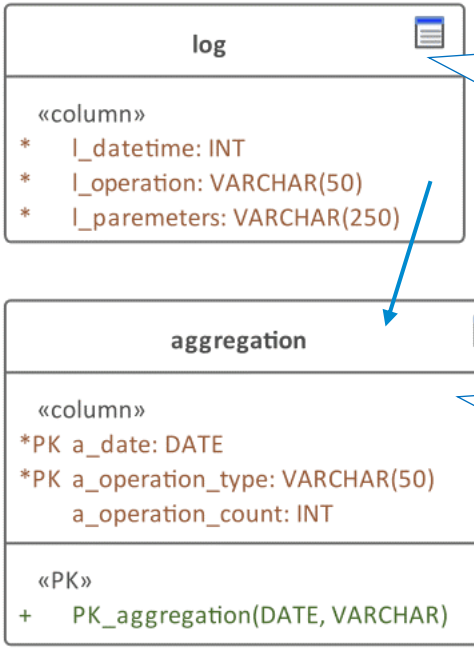
finance

…	f_serial_number	f_sum	…
…	AC345347856DF	29 627 532	…
…	DF345345652YH	12 000 000	…
…	AA345235235KL	32 511 012	…
…	GT456345342UT	41 356 343	…

General database performance

200-300K reads,
5-10 writes,
2-3 updates





0.1-0.2 reads,
500-700K writes,
0 updates

1-2 reads,
1 write,
0 updates

Data consistency and integrity

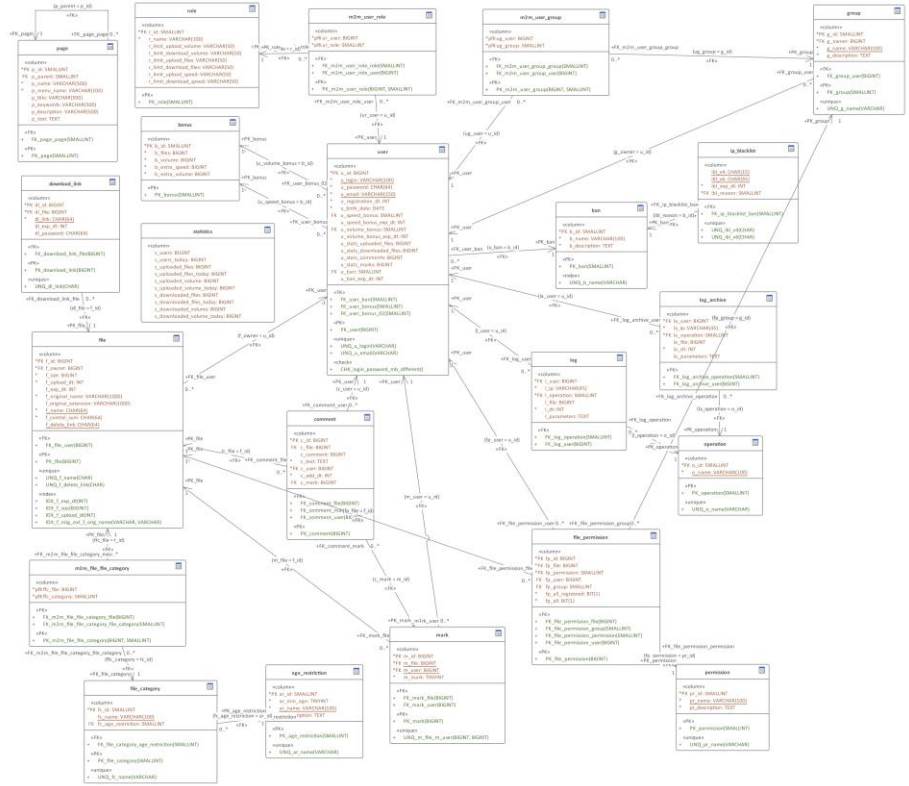
Create explicit relationships

Use triggers, checks, any other “constraint mechanisms”

Mind cache invalidation

All consistency and integrity constraints should be explicitly defined on database (NOT application) side!

Database flexibility through …



Database flexibility through naming convention

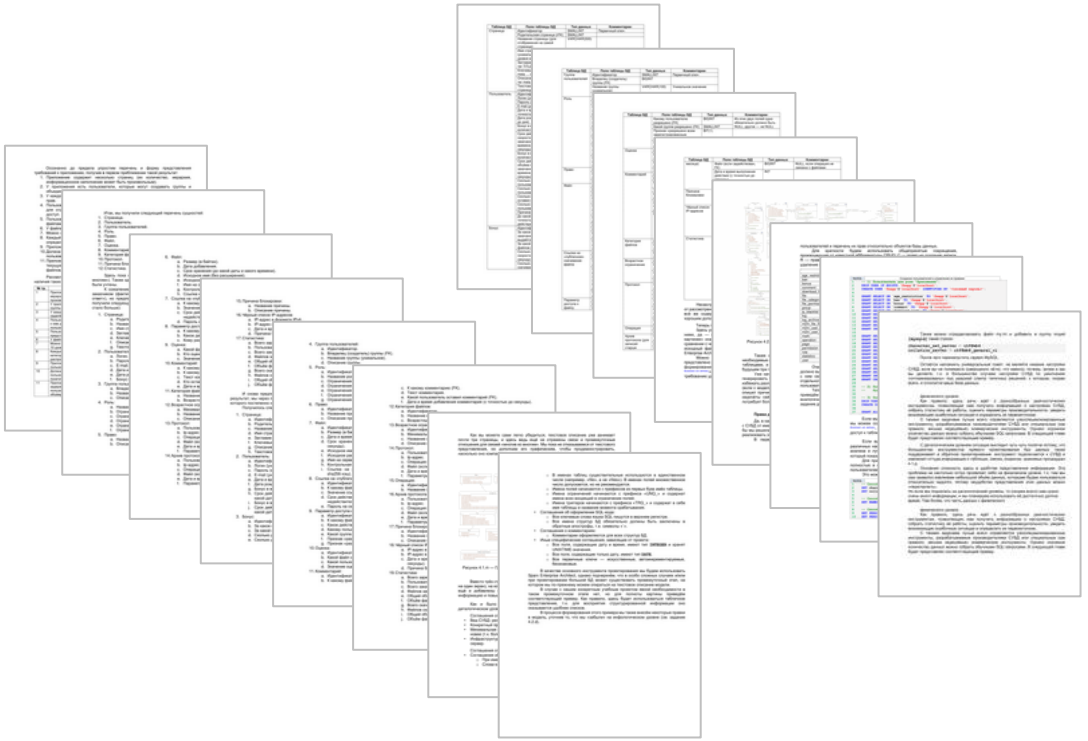
	BAD	GOOD
Table names	data	registered_user
	connection	m2m_user_role
	t1	news_rubric
Filed names	pp	u_primary_phone
	ok	o_is_order_confirmed
	some_text_data	a_biography
Stored subroutine names	process	get_initials
	get_date	unixtime_to_datetime
	fnc	upcase_first_letters
Trigger names	no_admins	t_user_protect_last_admin_del
	update_all	t_aggregate_user_stats_upd
	bad_dates	t_date_start_le_end_ins_upd
Index names	qsrch	idx_rubric_author_date
	singlerecord	unq_login
	tr	idx_author_btree

Database flexibility through comments

```
CREATE TABLE `file`
(
 `f_uid`              BIGINT UNSIGNED NOT NULL AUTO_INCREMENT
                      COMMENT 'Global file identifier.',
 `f_fc_uid`           BIGINT UNSIGNED NULL
                      COMMENT 'File category identifier (FK to "category" table).',
 `f_size`             BIGINT UNSIGNED NOT NULL
                      COMMENT 'File size (bytes).',
 `f_upload_datetime`  INTEGER NOT NULL
                      COMMENT 'File upload datetime (Unixtime).',
 `f_save_datetime`    INTEGER NOT NULL
                      COMMENT 'File expiration datetime (Unixtime).',
 `f_src_name`         VARCHAR(255) NOT NULL
                      COMMENT 'Initial file name (as it was on user device). With no
                               extension, as it is stored separately in "f_src_ext" field!',
 `f_src_ext`          VARCHAR(255) NULL
                      COMMENT 'Initial file extension (as it was on user device).',
 `f_name`             CHAR(200) NOT NULL
                      COMMENT 'Stored server-side file name. Five SHA1-hashes.',
 `f_sha1_checksum`    CHAR(40) NOT NULL
                      COMMENT 'File control sum, SHA1-hash.',
 `f_ar_uid`           BIGINT UNSIGNED NOT NULL
                      COMMENT 'File access permissions (FK to "access_rights" table).',
 `f_downloaded`       BIGINT UNSIGNED NULL DEFAULT 0
                      COMMENT 'File downloads counter.',
-- ...
```

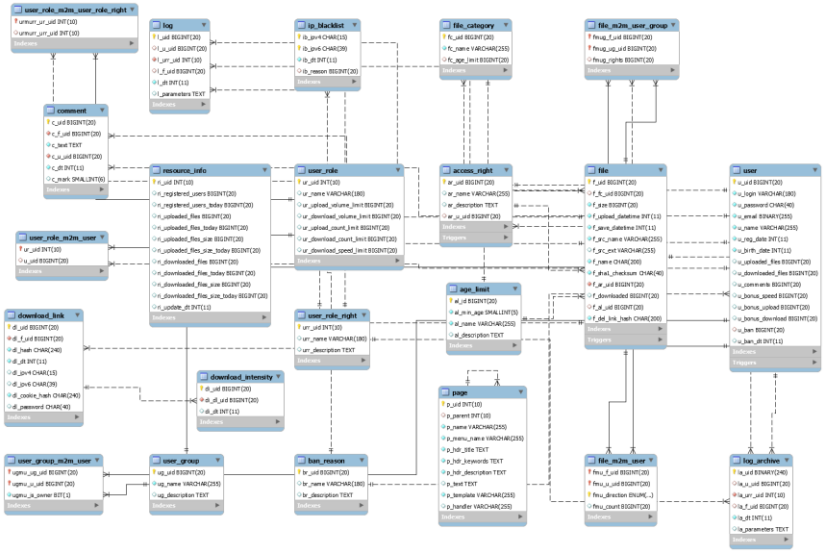
Database flexibility through documentation

Schemas and comments are good, but some text explanations are good too.

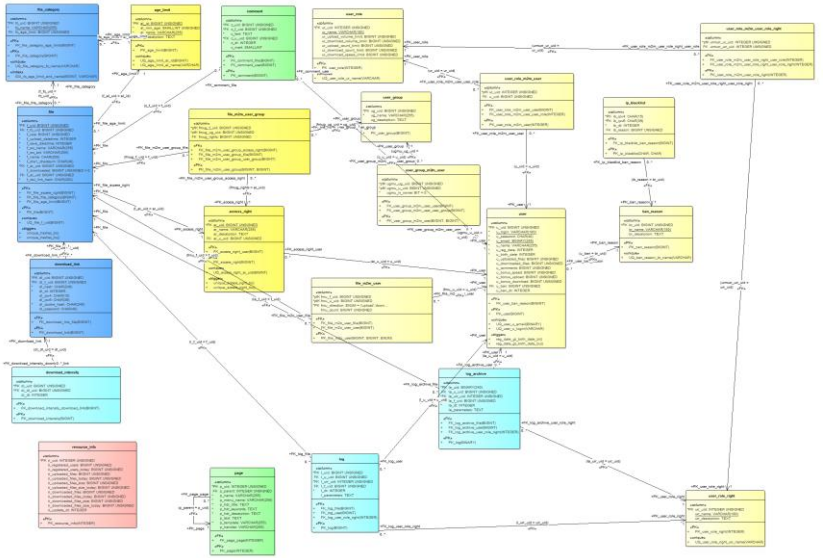


Database flexibility through schemas

Auto-generated



Manually created



Database flexibility through logical approach

Imagine you see such a code in database creation script…

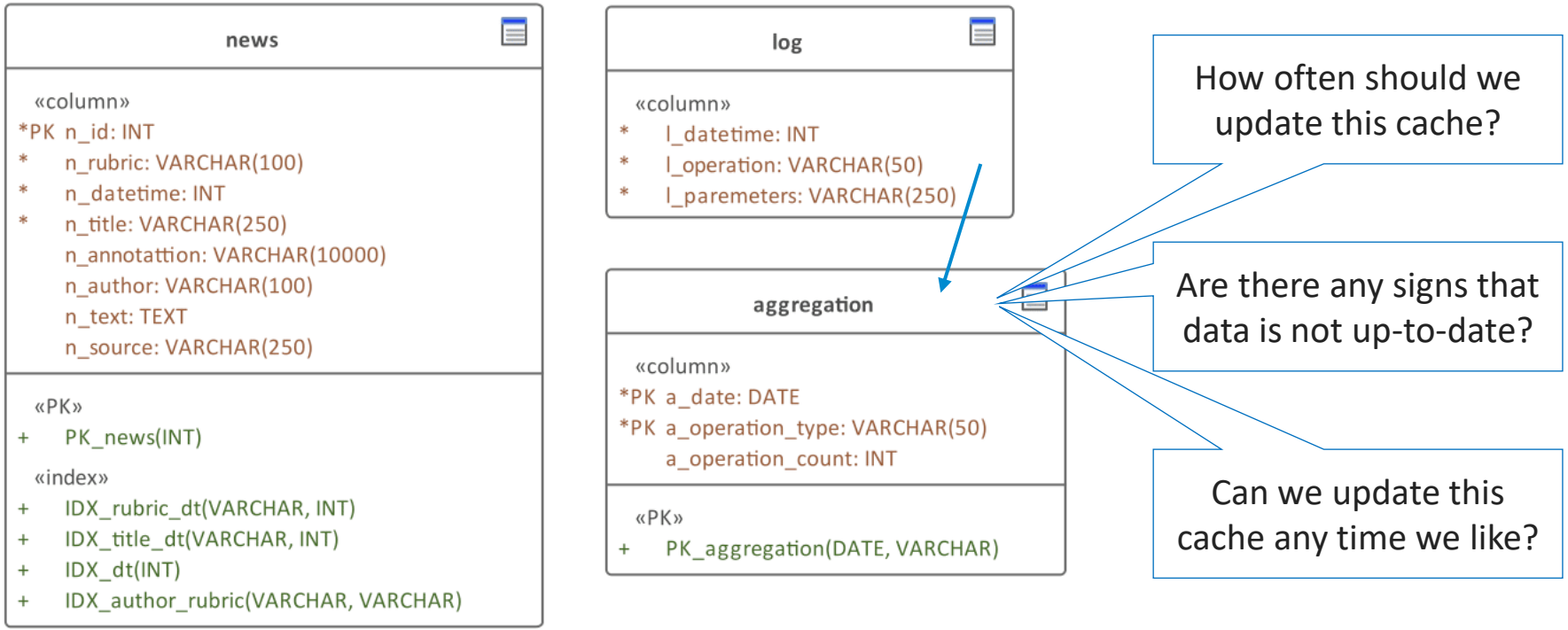
```
ALTER TABLE `file`
 ADD INDEX `IDX_f_orig_ext_f_orig_name`
  (`f_original_extension`(200) ASC,`f_original_name`(200) ASC);
```

???

???

Are these limitations necessary? Isn't it s mistake? Where are comments?!?!
Shall we leave this intact or change it? Will this change break anything?

Data relevance (up-to-date state)



<epam>

Normalization Requirements

Relational Databases Basics

TRAINING
CENTER

<epam>