# База данных электротехнической лаборатории.

1. **Краткое описание**

Предназначена для автоматизации работы лаборатории с помощью десктопного приложения на рабочем месте или мобильного приложения при выполнении измерении на объекте на территории заказчика.

Краткая схема бизнес-процесса:

1. Анализируют проект и составляют перечень оборудования, которое необходимо испытать.
2. Для каждой единицы оборудование изучают заводскую документацию и определяют список необходимых испытаний в соответствии с действующими нормами и требованиями заказчика.
3. Формируют состав бригады и необходимые им средства измерений (инструменты) для выполнения работы.
4. Выполняют измерения и испытания на объекте.
5. Анализируют полученные результаты и оформляют протокол испытаний.

Основной результат работы лаборатории – протокол испытаний. Разработанная база данных совместно с дополнительным программным обеспечением призвана автоматизировать, упростить и ускорить процесс создания протоколов тем самым сократить время выполнения работы и снизить вероятность ошибок, связанных с человеческим фактором при переносе данных, а также позволяет избежать потерю данных.

Условимся, что каждый протокол испытаний должен иметь уникальный номер, дату создания и содержать следующие данные:

1. Описание проекта.
2. Идентификация объекта испытаний с указание его основных параметров.
3. Перечень средств использованных средств измерений.
4. Результаты измерений и испытаний.
5. Перечень сотрудников, выполнявших измерения, а также оформивших и проверивших протокол.

Помимо этого, база данных может применятся для оценки статистики, связанной с результатами измерений, их количеством и точностью в зависимости от операторов, средств измерений или объектов испытаний.

1. **Таблицы**

2.1 Таблица с данными о сотрудниках лаборатории – personnel.

Содержит данные об имени, фамилии, отчестве и должности сотрудников.

Во избежание ошибочного дублирования данных не допускается добавлять в таблицу сотрудников с одинаковым полним именем. Используем уникальный индекс для ускорения поиска по имени.

**DROP** **TABLE** **IF** **EXISTS** personnel;

**CREATE** **TABLE** personnel(

id **INT** **UNSIGNED** **NOT** **NULL** **UNIQUE** **AUTO\_INCREMENT**,

surname **VARCHAR**(25) ,

name **VARCHAR**(25) **NOT** **NULL**,

patronymic **VARCHAR**(25) **NOT** **NULL**,

**position** **VARCHAR**(25) **NOT** **NULL**,

**UNIQUE** **INDEX** fullname (surname, name, patronymic)

);

2.2 Таблица с данными о средствах измерений лаборатории - instruments.

Содержит информацию о типе, модели и серийном номере инструмента, а также данные о дате последней метрологической оценки (калибровке или поверке) с указанием номера сертификата и сроком его действия.

Поля модель и номер сертификата не являются обязательными.

По умолчанию сертификат метрологической оценки выдается на год

Во избежание ошибочного дублирования данных не допускается добавлять в таблицу средства измерения с одинаковыми моделью и серийным номером.

Используем индекс для ускорения поиска по типу и модели.

**DROP** **TABLE** **IF** **EXISTS** instruments;

**CREATE** **TABLE** instruments(

id SERIAL,

**type** **VARCHAR**(50) **NOT** **NULL**,

model **VARCHAR**(25),

serial\_num **VARCHAR**(25) **NOT** **NULL**,

calibration\_sertificate **VARCHAR**(25),

calibration\_date **DATE** **NOT** **NULL** **DEFAULT** (**CURRENT\_DATE**),

calibration\_validity **DATE** **NOT** **NULL** **DEFAULT** (calibration\_date + **INTERVAL** 1 **YEAR**),

**UNIQUE** **KEY** item (model, serial\_num),

**INDEX** (**type**, model)

);

2.3 Таблица с данными о проектах, в которых участвует лаборатория - projects.

Содержит информацию заказчике, названии и адресе объекта, а также поля с указанием номера и датой договора, которые являются необязательными.

Используем отдельные индексы для ускорения поиска по заказчику и по номеру договора.

**DROP** **TABLE** **IF** **EXISTS** projects;

**CREATE** **TABLE** projects(

id SERIAL,

customer **VARCHAR**(100) **NOT** **NULL**,

title **TEXT** **NOT** **NULL**,

address **VARCHAR**(100) **NOT** **NULL**,

contract\_num **VARCHAR** (15),

contract\_date **DATETIME**,

**INDEX** (customer),

**INDEX** (contract\_num)

);

2.4 Таблица с регистрационными данными протоколов испытаний – reports.

Таблица с базовыми данными протоколов испытаний. Содержит номер протокола, название и дату его создания.

Дополнительное поле определяет год создания протокола для однозначной идентификации и сортировки по годам. Во избежание ошибочного внесения неверного значения в это поле в ручную используем дополнительную проверку.

Таблица включает поле ссылающееся на id таблицы projects для отнесения протокола к одному из проектов лаборатории (связь один ко многим с обязательной связью).

Таблица включает два поля ссылающееся на id таблицы personnel для указания сотрудников, оформивших и проверивших протокол (связь один ко многим с обязательной связью). По умолчанию протоколы проверяет начальник лаборатории – сотрудник с id = 1.

Следует отметить, что созданные связи не позволят удалить записи в таблицах projects и personnel, если они уже используются в таблице reports, что обеспечивает сохранность и непрерывность данных при длительной эксплуатации базы.

**DROP** **TABLE** **IF** **EXISTS** reports;

**CREATE** **TABLE** reports(

num **INT** **UNSIGNED** **NOT** **NULL** **DEFAULT** 0,

creation\_date **DATE** **NOT** **NULL** **DEFAULT** (**CURRENT\_DATE**),

creation\_year **YEAR** **NOT** **NULL**,

title **VARCHAR**(50),

project\_id **BIGINT** **UNSIGNED** **NOT** **NULL**,

edited\_by **INT** **UNSIGNED** **NOT** **NULL**,

checked\_by **INT** **UNSIGNED** **NOT** **NULL** **DEFAULT** 1,

**FOREIGN** **KEY** (project\_id) **REFERENCES** projects (id),

**FOREIGN** **KEY** (edited\_by) **REFERENCES** personnel (id),

**FOREIGN** **KEY** (checked\_by) **REFERENCES** personnel (id),

**PRIMARY** **KEY** pk (num, creation\_year),

**INDEX** (creation\_year, num)

);

2.5 Таблицы для хранения характеристик проверяемого оборудования.

В процессе деятельности испытательной лаборатории проверкам подвергается электротехническое оборудование разного типа и разных категорий: силовые и измерительные трансформаторы, выключатели, силовые кабели, распределительные устройства и многое другое. Каждое из них имеет свой набор характеристик в зависимости от типа, производителя и индивидуальных особенностей.

Для организации универсальной структуры хранения характеристик используем структуру EAV (Entity-attribute-value). «Сущностями» в данном случае выступают образцы проверяемого оборудования, которые хранятся в таблице equipment. Предварительно разобьём проверяемое оборудование на категории, которые будем хранить в таблице equipment\_categories и определим перечень атрибутов, относящихся к этим категорям. Такой шаг позволит создавать функции для унифицированного создания объектов с базовыми атрибутами, однако не запрещает использование дополнительных атрибутов для конкретного оборудования, если это необходимо.

2.5.1 Таблица-каталог с категориями проверяемого оборудования - equipment\_categories.

**DROP** **TABLE** **IF** **EXISTS** equipment\_categories;

**CREATE** **TABLE** equipment\_categories(

id **INT** **UNSIGNED** **NOT** **NULL** **AUTO\_INCREMENT** **PRIMARY** **KEY**,

title **VARCHAR**(50) **NOT** **NULL** **UNIQUE**

);

2.5.2 Таблица с наименованием проверяемого оборудования - equipment.

Содержит ссылки на категорию оборудования (связь один ко многим без обязательной связи) и на проект, в котором это оборудование используется (связь один ко многим с обязательной связью).

Удаление записи из таблицы equipment\_categories не возможно, если уже добавлено оборудование этой категории.

Удаление записи из таблицы projects приведет к удалению всего оборудования, относящегося к проекту.

Используем отдельные индексы для ускорения поиска по проекту и категории оборудования.

**DROP** **TABLE** **IF** **EXISTS** equipment;

**CREATE** **TABLE** equipment(

id SERIAL,

title **VARCHAR**(50) **NOT** **NULL**,

category\_id **INT** **UNSIGNED**,

project\_id **BIGINT** **UNSIGNED** **NOT** **NULL**,

**FOREIGN** **KEY** (category\_id) **REFERENCES** equipment\_categories (id),

**FOREIGN** **KEY** (project\_id) **REFERENCES** projects (id) **ON** **DELETE** **CASCADE**,

**INDEX** (project\_id),

**INDEX** (category\_id)

);

2.5.3 Таблица с атрибутами проверяемого оборудования по категориям.

Содержит ссылку на таблицу equipment\_categories (связь один ко многим без обязательной связи), при удалении записи из которой, удаляются все атрибуты относящиеся к этой категории.

Запись также может не относиться ни к одной из категорий.

Во избежание ошибочного дублирования данных не допускается добавлять в таблицу атрибуты с одинаковыми названиями, относящиеся к одной и той же категории оборудования.

**DROP** **TABLE** **IF** **EXISTS** equipment\_attributes;

**CREATE** **TABLE** equipment\_attributes(

id SERIAL,

category\_id **INT** **UNSIGNED**,

title **VARCHAR**(50) **NOT** **NULL**,

**FOREIGN** **KEY** (category\_id) **REFERENCES** equipment\_categories (id) **ON** **DELETE** **CASCADE**,

**UNIQUE** **KEY** (category\_id, title)

)

2.5.4 Таблица с характеристиками проверяемого оборудования - equipment\_values.

Содержит ссылки на объекты оборудования и атрибуты (связь один ко многим с обязательной связью), при удалении которых удаляются и записи в таблице со значениями.

Используем составной primary key вместо id, поскольку этого достаточно для обеспечения уникальности записей, и, маловероятно, что пользователей будет интересовать лишь одна конкретная характеристика.

**DROP** **TABLE** **IF** **EXISTS** equipment\_values;

**CREATE** **TABLE** equipment\_values(

equipment\_id **BIGINT** **UNSIGNED** **NOT** **NULL**,

attribute\_id **BIGINT** **UNSIGNED** **NOT** **NULL**,

value **VARCHAR**(50),

**PRIMARY** **KEY** (equipment\_id, attribute\_id),

**FOREIGN** **KEY** (equipment\_id) **REFERENCES** equipment (id) **ON** **DELETE** **CASCADE**,

**FOREIGN** **KEY** (attribute\_id) **REFERENCES** equipment\_attributes (id) **ON** **DELETE** **CASCADE**

);

2.6 Таблица-каталог с наименованием испытаний, выполняемых лабораторией - tests.

Содержит название теста и идентификатор измеряемого параметра, а также может содержать размерность измеряемого параметра.

Во избежание ошибочного дублирования данных не допускается добавлять в таблицу записи с одинаковыми названием теста и идентификатором измеряемого параметра.

**DROP** **TABLE** **IF** **EXISTS** tests;

**CREATE** **TABLE** tests(

id **INT** **UNSIGNED** **NOT** **NULL** **AUTO\_INCREMENT** **PRIMARY** **KEY**,

test **VARCHAR**(100) **NOT** **NULL**,

**parameter** **VARCHAR**(100) **NOT** **NULL**,

dimension **VARCHAR**(10),

**UNIQUE** **KEY** (test, **parameter**)

);

2.7 Таблица с результатами выполненных измерений - results.

Содержит ссылки на объект испытаний, тип испытаний и средство измерений (связи один к многим с обязательной связью).

При удалении объекта испытаний записи с результатами измерений, относящиеся к объекту также удаляются.

Удаление записей из таблиц tests и instruments не возможно, если они уже используются в результатах измерений.

Содержит режим измерений, результат, размерность измеряемой величины, дату измерений и заметку.

Во избежание ошибочного дублирования данных используем сложный составной unique key.

**DROP** **TABLE** **IF** **EXISTS** results;

**CREATE** **TABLE** results(

id SERIAL,

equipment\_id **BIGINT** **UNSIGNED** **NOT** **NULL**,

test\_id **INT** **UNSIGNED** **NOT** **NULL**,

instrument\_id **BIGINT** **UNSIGNED** **NOT** **NULL**,

mode **VARCHAR**(25) **NOT** **NULL** **DEFAULT** '',

**result** **FLOAT** **UNSIGNED** **NOT** **NULL**,

dimension **VARCHAR**(10),

performed\_on **DATE** **NOT** **NULL** **DEFAULT** (**CURRENT\_DATE**),

remark **VARCHAR**(250) **NOT** **NULL** **DEFAULT** '',

**UNIQUE** **KEY** (equipment\_id, test\_id, mode, performed\_on, remark), -- если измерение повторялось несколько раз в течение дня, необходимо сделать пометку в remark для идентификации

**FOREIGN** **KEY** (equipment\_id) **REFERENCES** equipment(id) **ON** **DELETE** **CASCADE**,

**FOREIGN** **KEY** (test\_id) **REFERENCES** tests(id),

**FOREIGN** **KEY** (instrument\_id) **REFERENCES** instruments(id)

);

2.8 Таблица с данными о том, кто из сотрудников участвовал в получении результатов - participants.

Таблица обеспечивает связь многие ко многим, поскольку каждое измерение выполняет группа сотрудников, при этом каждый сотрудник задействован в большом количестве измерений.

Вместо id используем составной primary key.

При удалении записи из таблицы results удаляются и соответствующие ей записи в таблице participants.

Удаление записи из таблицы personnel не возможно, если эта запись связана с таблицей participants.

**DROP** **TABLE** **IF** **EXISTS** participants;

**CREATE** **TABLE** participants(

results\_id **BIGINT** **UNSIGNED** **NOT** **NULL**,

personnel\_id **INT** **UNSIGNED** **NOT** **NULL**,

**PRIMARY KEY** (results\_id, personnel\_id),

**FOREIGN** **KEY** (results\_id) **REFERENCES** results (id) **ON** **DELETE** **CASCADE**,

**FOREIGN** **KEY** (personnel\_id) **REFERENCES** personnel (id),

**INDEX** (results\_id)

);

2.9 Таблица с данными о том, какие результаты включаются в тот или иной протокол - report\_results.

Таблица обеспечивает связь многие ко многим, поскольку каждый протокол включает несколько измерений и, при этом, каждое измерение может быть включено в несколько протоколов.

Вместо id используем составной primary key.

При удалении записи из таблицы reports удаляются и соответствующие ей записи в таблице report\_results.

Удаление записи из таблицы results не возможно, если эта запись связана с таблицей report\_results.

**DROP** **TABLE** **IF** **EXISTS** report\_results;

**CREATE** **TABLE** report\_results(

report\_num **INT** **UNSIGNED** **NOT** **NULL**,

results\_id **BIGINT** **UNSIGNED** **NOT** **NULL**,

**UNIQUE** **KEY** (report\_num, results\_id),

**FOREIGN** **KEY** (report\_num) **REFERENCES** reports (num) **ON** **DELETE** **CASCADE**,

**FOREIGN** **KEY** (results\_id) **REFERENCES** results (id),

**INDEX** (report\_num)

);

1. **Триггеры**

3.1 Измененный autoincrement в таблице reports.

На первом этапе заполним поле creation\_year на основании данных из поля creation\_date. Далее переопределим autoincrement так, чтобы оставалась возможность заполнять пропущенные номера протоколов.

**DELIMITER //**

**DROP** **TRIGGER** **IF** **EXISTS** autoincrement\_mod//

**CREATE** **TRIGGER** autoincrement\_mod

**BEFORE** **INSERT** **ON** reports

**FOR** **EACH** **ROW**

**BEGIN**

**DECLARE** bigest\_num **INT**;

**SET** **NEW**.creation\_year = **YEAR**(**NEW**.creation\_date);

**IF** **NEW**.num = 0 **THEN**

**BEGIN**

**SET** bigest\_num = **COALESCE**((**SELECT** **MAX**(num) **FROM** reports **WHERE** creation\_year = **NEW**.creation\_year), 0);

**SET** **NEW**.num = bigest\_num + 1;

**END**;

**END** **IF**;

**END**//

**DELIMITER ;**

3.2 Триггер end\_to\_end\_numbering для таблицы reports.

Для обеспечения сквозной нумерации протоколов триггер end\_to\_end\_numbering проверяет, чтобы протоколы с большим номером не был создан раньше протокола с меньшим номером.

**DELIMITER //**

**DROP** **TRIGGER** **IF** **EXISTS** end\_to\_end\_numbering\_insert//

**CREATE** **TRIGGER** end\_to\_end\_numbering\_insert **BEFORE** **INSERT** **ON** reports

**FOR** **EACH** **ROW**

**BEGIN**

**DECLARE** previous\_date, next\_date **DATE**;

**SET** previous\_date = (**SELECT** **MAX**(creation\_date) **FROM** reports

**WHERE** creation\_year = **NEW**.creation\_year

**AND** num < **NEW**.num);

**IF** previous\_date **IS** **NOT** **NULL** **AND** **NEW**.creation\_date < previous\_date

**THEN** **SIGNAL** **SQLSTATE** '45000' **SET** MESSAGE\_TEXT = "Ошибка! Протокол с меньшим номером уже зарегистрирован на более позднюю дату.";

**END** **IF**;

**SET** next\_date = (**SELECT** **MIN**(creation\_date) **FROM** reports

**WHERE** creation\_year = **NEW**.creation\_year

**AND** num > **NEW**.num);

**IF** next\_date **IS** **NOT** **NULL** **AND** **NEW**.creation\_date > next\_date

**THEN** **SIGNAL** **SQLSTATE** '45000' **SET** MESSAGE\_TEXT = "Ошибка! Протокол с большим номером уже зарегистрирован на более раннюю дату.";

**END** **IF**;

**END**//

**DROP** **TRIGGER** **IF** **EXISTS** end\_to\_end\_numbering\_update//

**CREATE** **TRIGGER** end\_to\_end\_numbering\_update **BEFORE** **UPDATE** **ON** reports

**FOR** **EACH** **ROW**

**BEGIN**

**DECLARE** previous\_date, next\_date **DATE**;

**SET** previous\_date = (**SELECT** **MAX**(creation\_date) **FROM** reports

**WHERE** creation\_year = **NEW**.creation\_year

**AND** num < **NEW**.num);

**IF** previous\_date **IS** **NOT** **NULL** **AND** **NEW**.creation\_date < previous\_date

**THEN** **SIGNAL** **SQLSTATE** '45000' **SET** MESSAGE\_TEXT = "Ошибка! Протокол с меньшим номером уже зарегистрирован на более позднюю дату.";

**END** **IF**;

**SET** next\_date = (**SELECT** **MIN**(creation\_date) **FROM** reports

**WHERE** creation\_year = **NEW**.creation\_year

**AND** num > **NEW**.num);

**IF** next\_date **IS** **NOT** **NULL** **AND** **NEW**.creation\_date > next\_date

**THEN** **SIGNAL** **SQLSTATE** '45000' **SET** MESSAGE\_TEXT = "Ошибка! Протокол с большим номером уже зарегистрирован на более раннюю дату.";

**END** **IF**;

**END**//

**DELIMITER ;**

3.3 Триггер result\_demension для таблицы results.

При создании записи позволяет не указывать размерность измеряемой величины. При это в поле dimension записывается базовое значение из таблицы tests.

**DELIMITER //**

**CREATE** **TRIGGER** result\_demension **BEFORE** **INSERT** **ON** results

**FOR** **EACH** **ROW**

**BEGIN**

**IF** **NEW**.dimension **IS** **NULL** **THEN**

**SET** **NEW**.dimension = (**SELECT** dimension **FROM** tests **WHERE** id = **NEW**.test\_id);

**END** **IF**;

**END**//

**DELIMITER ;**

1. **Характерные выборки**

4.1 Запрос на получение характеристик оборудования с id=1.

**SELECT** ea.title, ev.value **FROM** equipment\_values ev

**JOIN** equipment\_attributes ea **ON** ev.attribute\_id = ea.id

**WHERE** equipment\_id = 1;

4.2 Запрос на получение всех результатов, которые необходимо внести в протокол num = 25

**SELECT** t.test, r.mode, t.**parameter**, r.**result**, r.dimension

**FROM** results r

**JOIN** tests t **ON** r.test\_id = t.id

**JOIN** report\_results rr **ON** r.id = rr.results\_id

**WHERE** rr.report\_num = 25

**ORDER** **BY** t.test, r.mode;

4.3 Запрос на получение средств измерений, использованных для получения результатов, внесенных в протокол num = 25.

**SELECT** **DISTINCT** i.**type**, i.model, i.serial\_num, i.calibration\_validity

**FROM** results r

**JOIN** report\_results rr **ON** r.id = rr.results\_id

**JOIN** instruments i **ON** r.instrument\_id = i.id

**WHERE** rr.report\_num = 25;

4.4 Запрос на получение сотрудников, участвовавших в получении результатов, внесенных в протокол num = 25.

**SELECT** **DISTINCT**

**CONCAT**(p2.surname, ' ', **SUBSTRING**(p2.name, 1, 1), '.', **SUBSTRING**(p2.patronymic, 1, 1), '.') **AS** name,

p2.**position**

**FROM** results r

**JOIN** report\_results rr **ON** r.id = rr.results\_id

**JOIN** participants p **ON** r.id = p.results\_id

**JOIN** personnel p2 **ON** p2.id = p.personnel\_id

**WHERE** rr.report\_num = 25;

1. **Представления**

5.1 Выборка результатов измерений, которые выполнил сотрудник с указанием id сотрудников, выполнявших измерения.

Такое представление с параметром ALGORITHM = MERGE необходимо для предоставления права на редактирования результатов, только сотрудником, получившим эти результаты.

**CREATE** **OR** **REPLACE** **ALGORITHM** = **MERGE** **VIEW** results\_by\_personnel\_view **AS**

**SELECT** r.\*, p.personnel\_id **FROM** results r, participants p

**WHERE** r.id = p.results\_id;

5.2 Представления report\_results\_view, report\_instruments\_view и report\_personnel\_view созданы на основании типовых запросов из пунктов 4.2 – 4.4 для более простого получения соответствующих данных.

**CREATE** **OR** **REPLACE** **VIEW** report\_results\_view **AS**

**SELECT** rr.report\_num, t.test, r.mode, t.**parameter**, r.**result**, r.dimension

**FROM** results r, tests t, report\_results rr

**WHERE** r.test\_id = t.id **AND** r.id = rr.results\_id

**ORDER** **BY** rr.report\_num, t.test, r.mode;

**SELECT** test, mode, **parameter**, **result**, dimension

**FROM** (**SELECT** \* **FROM** report\_results\_view **HAVING** report\_num = 25) **AS** rrv;

**CREATE** **OR** **REPLACE** **VIEW** report\_instruments\_view **AS**

**SELECT** **DISTINCT** rr.report\_num, i.**type**, i.model, i.serial\_num, i.calibration\_validity

**FROM** results r, report\_results rr, instruments i

**WHERE** r.id = rr.results\_id **AND** r.instrument\_id = i.id;

**SELECT** **DISTINCT** **type**, model, serial\_num, calibration\_validity

**FROM** (**SELECT** \* **FROM** report\_instruments\_view **HAVING** report\_num = 25) **AS** riv;

**CREATE** **OR** **REPLACE** **VIEW** report\_personnel\_view **AS**

**SELECT** **DISTINCT**

rr.report\_num,

**CONCAT**(p2.surname, ' ', **SUBSTRING**(p2.name, 1, 1), '.', **SUBSTRING**(p2.patronymic, 1, 1), '.') **AS** name,

p2.**position**

**FROM** results r, report\_results rr, participants p, personnel p2

**WHERE** r.id = rr.results\_id **AND** r.id = p.results\_id **AND** p2.id = p.personnel\_id;

**SELECT** **DISTINCT** name, **position**

**FROM** (**SELECT** \* **FROM** report\_personnel\_view **HAVING** report\_num = 25) **AS** riv;