

3. Введение в SQL.

к.т.н., доцент кафедры ИиСП

Лучинин
Захар Сергеевич

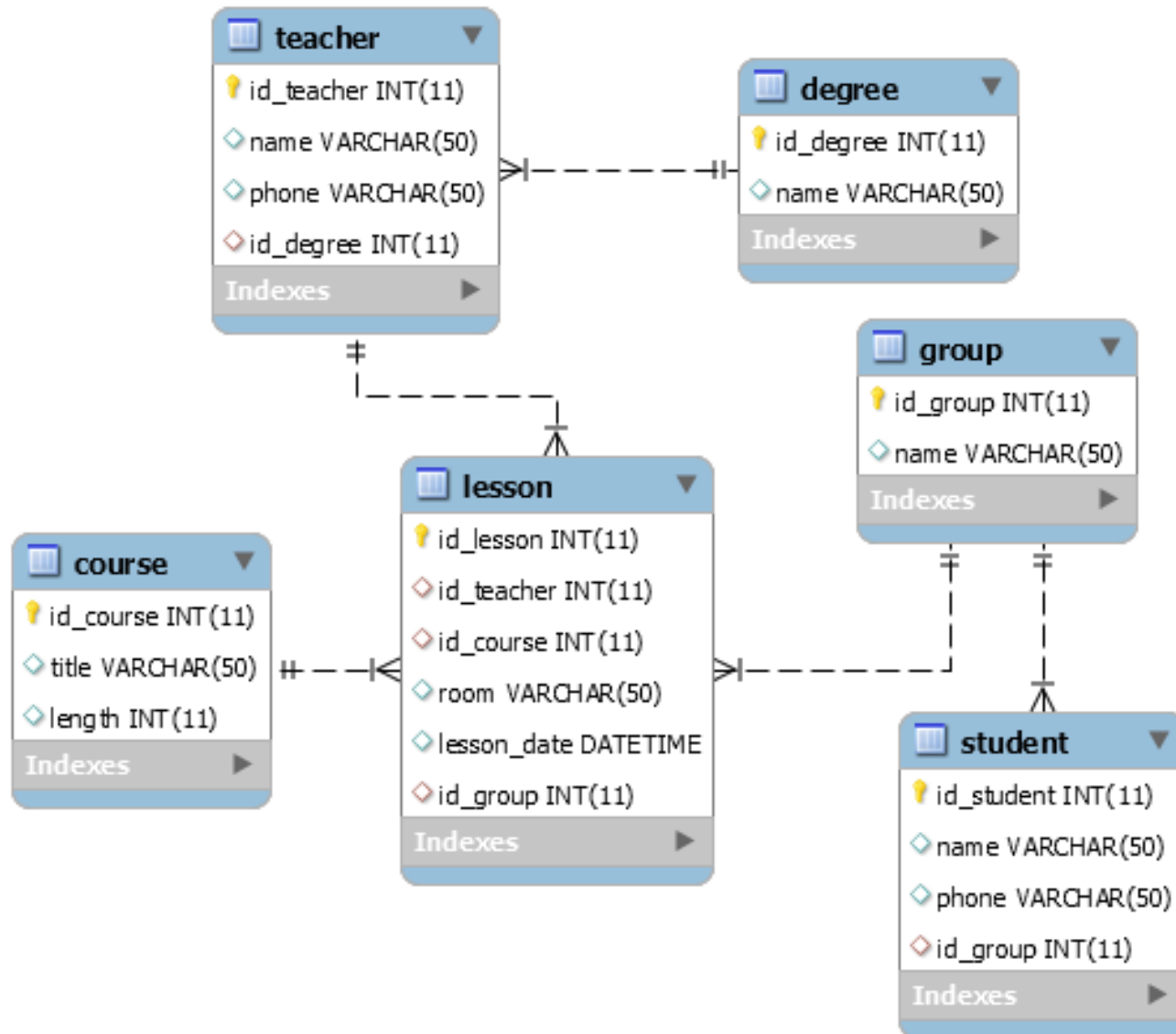
Уровни проектирования

1. Концептуальное (инфологическое) проектирование
2. Логическое (дatalogическое) проектирование
3. **Физическое проектирование**

Процедуры физического проектирования

1. Проектирование таблиц базы данных средствами выбранной СУБД.
2. Реализация бизнес-правил в среде выбранной СУБД.
3. Проектирование физической организации базы данных.
4. Разработка стратегии защиты базы данных.
5. Организация мониторинга функционирования базы данных и ее настройка.

ER диаграмма физического уровня проектирования



SQL

- **SQL** - (structured query language — «язык структурированных запросов») язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных.
 - создание в базе данных новой таблицы;
 - добавление в таблицу новых записей;
 - изменение записей;
 - удаление записей;
 - выборка записей из одной или нескольких таблиц (в соответствии с заданным условием);
 - изменение структур таблиц.

Дефекты SQL от Эдгара Кодда

- SQL разрешает в таблицах строки-дубликаты, что в рамках реляционной модели данных невозможно и недопустимо;
- SQL поддерживает неопределённые значения (NULL) и многозначную логику;
- SQL использует порядок колонок и ссылки на колонки по номерам;
- SQL разрешает колонки без имени и дублирующиеся имена колонок.

SQL стандарт

Год	Название	Иное название	Изменения
1986	SQL-86	SQL-87	Первый вариант стандарта, принятый институтом ANSI и одобренный ISO в 1987 году.
1989	SQL-89	FIPS 127-1	Немного доработанный вариант предыдущего стандарта.
1992	SQL-92	SQL2, FIPS 127-2	Значительные изменения (ISO 9075); уровень Entry Level стандарта SQL-92 был принят как стандарт FIPS 127-2.
1999	SQL:1999	SQL3	Добавлена поддержка регулярных выражений, рекурсивных запросов, поддержка триггеров, базовые процедурные расширения, нескалярные типы данных и некоторые объектно-ориентированные возможности.
2003	SQL:2003		Введены расширения для работы с XML-данными
2006	SQL:2006		Функциональность работы с XML-данными значительно расширена.
2008	SQL:2008		Улучшены возможности оконных функций, устранены некоторые неоднозначности стандарта SQL:2003[7]
2011	SQL:2011		Реализована поддержка хронологических баз данных (PERIOD FOR), поддержка конструкции FETCH ^[8] .
2016	SQL:2016		Защита на уровне строк, полиморфные табличные функции, JSON.

SQL диалекты

Источник	Аббревиатура	Полное название
ANSI/ISO Standard	SQL/PSM	SQL/Persistent Stored Modules
Microsoft / Sybase	T-SQL	Transact-SQL
MySQL	SQL/PSM	SQL/Persistent Stored Module (implements SQL/PSM)
Oracle	PL/SQL	Procedural Language/SQL (based on Ada)
PostgreSQL	PL/pgSQL	Procedural Language/PostgreSQL Structured Query Language (implements SQL/PSM)

Несмотря на наличие международного стандарта ANSI SQL-92, многие разработчики СУБД отступают от стандарта. Таким образом появляются специфичные для каждой конкретной СУБД диалекты языка SQL.

Структура SQL

DDL

(Data Definition Language) – работа со структурой базы

- CREATE
- ALTER
- DROP
- TRUNCATE

DML

(Data Manipulation Language) – работа с данными

- SELECT
- INSERT
- UPDATE
- DELETE

DCL

(Data Control Language) – работа с правами

- GRANT
- REVOKE
- DENY

TCL

(Transaction Control Language) – работа с транзакциями

- BEGIN TRANSACTION
- COMMIT
- ROLLBACK
- SAVEPOINT

Понятие типа данных

Тип данных характеризует одновременно:

- множество допустимых значений, которые могут принимать данные, принадлежащие к этому типу;
- набор операций, которые можно осуществлять над данными, принадлежащими к этому типу.

Целые числа (Точные числовые типы)

Тип	От	До	Байт
bigint	-9,223,372,036,854,775,808	9,223,372,036,854,775,807	8
int	-2,147,483,648	2,147,483,647	4
smallint	-32,768	32,767	2
tinyint	-128	127	1
bit	0	1	1
decimal(P, S) / numeric (P, S)	$-10^{38} + 1$	$10^{38} - 1$	5-17
money	-922,337,203,685,477.5808	+922,337,203,685,477.5807	8
smallmoney	-214,748.3648	+214,748.3647	4

Дробные числа (Приблизительные числовые типы)

Тип	От	До
float(N)	-1.79E + 38	1.79E + 38
real	-3.40E + 38	3.40E + 38

Синонимом по стандарту ISO для типа real является float(24).

N Значение	Точность	Объем памяти
1-24	7 знаков	4 байт
25-53	15 знаков	8 байт

N – это количество битов, используемых для хранения мантиссы числа в формате float при экспоненциальном представлении. Значение параметра n должно лежать в пределах от 1 до 53. Значением по умолчанию для параметра n является 53.

Модификатор UNSIGNED

- **unsigned** - модификатор беззнакового типа данных
- Применение гарантирует, что переменная никогда не станет отрицательной.
- Данные указанного типа могут принимать большие значения, чем данные эквивалентного типа со знаком.

Рецепты по выбору типа данных

- Используй **UNSIGNED** если число без знака
- Хранение bool – **TINYINT** (bit в T-SQL)
- Идентификатор/первичный ключ – **UNSIGNED INT**
- Перечислимый тип данных – **TINYINT**
- Не вводи в заблуждение - **SMALLINT**

Символьные типы данных

- Типы данных **CHAR** и **VARCHAR**
- Типы данных **BLOB** и **TEXT**
- Тип перечисления **ENUM**
- Тип множества **SET**

Типы данных CHAR и VARCHAR

Величина	CHAR(4)	Требуемая память	VARCHAR(4)	Требуемая память
"	"	4 байта	"	1 байт
'ab'	'ab'	4 байта	'ab'	3 байта
'abcd'	'abcd'	4 байта	'abcd'	5 байтов
'abcdefgh'	'abcd'	4 байта	'abcd'	5 байтов

Типы данных BLOB и TEXT

Тип	Макс.размер	Байт
TINYTEXT TINYBLOB	2^8-1	255
TEXT BLOB	$2^{16}-1$ (64K-1)	65535
MEDIUMTEXT MEDIUMBLOB	$2^{24}-1$ (16M-1)	16777215
LOB LONGBLOB	$2^{32}-1$ (4G-1)	4,29E+09

Тип перечисления ENUM

Тип	Макс.размер	Байт
ENUM	65535 вариантов	2

- ENUM (перечисление) - это столбец, который может принимать значение из списка допустимых значений, явно перечисленных в спецификации столбца в момент создания таблицы.

Тип множества SET

Тип	Макс.размер	Байт
SET	64 элемента	8

- SET - это строковый тип, который может принимать ноль или более значений, каждое из которых должно быть выбрано из списка допустимых значений, определенных при создании таблицы. Элементы множества SET разделяются запятыми. Как следствие, сами элементы множества не могут содержать запятых.

Например, столбец, определенный как SET("один", "два") NOT NULL может принимать такие значения:

```
"" "один" "два" "один, два"
```

Рецепты по выбору типа данных

- Не используй **ENUM**, используй **TINYINT**
- Не используй **SET**, используй bit flags
- Используй **VARCHAR (255)** (nvarchar(4000) в T-SQL)
- Не забывай про кодировку
- Файлы в **BLOB** не храним

Временные типы данных

Тип	Диапазон	Требуемая память
DATE	от 1001-01-01 до 9999-12-31	3 байта
DATETIME	от 1001-01-01 00:00:00 до 9999-12-31 23:59:59	8 байт
TIME	от -838:59:59 до 838:59:59	3 байта
TIMESTAMP	от 1970-01-01 00:00:00 до 2037	4 байта
YEAR	от 1900 до 2155 от 1970 до 2069	1 байт

Рецепты по выбору типа данных

- Используй **TIMESTAMP** если необходимо смещение от 1970-01-01
- Используй **DATE** для хранения точки отсчета, а не текущий возраст

Значение по умолчанию

- Может быть задано значение по умолчанию, т.е. значение, которое будет подставляться в том случае, когда оператор вставки не предоставляет значения для этого столбца
- Значение определяется в момент создания атрибута, например, подстановку текущей даты или timestamp

NULL-значения

Введено с целью различать в полях БД пустые (визуально не отображаемые) значения (например, строку нулевой длины) и отсутствующие значения.

Если значение по умолчанию не задано и пользователь оставляет столбец пустым, происходит следующее:

- если активирована поддержка значений NULL, в столбец вставляется значение NULL;
- если поддержка значений NULL не активирована, столбец остается пустым, но пользователь не сможет сохранить строку, пока не предоставит какое-либо значение.

AUTO_INCREMENT

IDENTITY

- Используется для генерации уникального идентификатора для новых записей БД
- **Суррогатный ключ** — дополнительное служебное поле, добавленное к уже имеющимся информационным полям таблицы, единственное предназначение которого — служить первичным ключом.

Автоинкремент vs GUID

Автоинкремент

- Занимает меньший объем
- Теоретически, более быстрая генерация нового значения
- Более быстрая десериализация
- Проще оперировать при отладке, поддержке, так как число гораздо легче запомнить

GUID

- При репликации между несколькими экземплярами базы, где добавление новых записей происходит более, чем в одну реплику, GUID гарантирует отсутствие коллизий
- Позволяет генерировать идентификатор записи на клиенте, до сохранения ее в базу
- Обобщение первого пункта — обеспечивает уникальность идентификаторов не только в пределах одной таблицы, что для некоторых решений может быть важно
- Делает практически невозможным «угадывание» ключа в случаях, когда запись можно получить, передав ее идентификатор в какой-нибудь публичный API

Пример использования уникального идентификатора

marimedia.ru/poster/10313/

Афиша

- Все события 146
- Кино 12
- Театры 47
- Клубы и бары 2
- Концерты 16
- Выставки 19
- Детям 41
- Город 8
- Мастер-классы 1

Архив событий

+ Предложить событие

Би-2

22 МАРТА, 19:00

Ледовый дворец «Марий Эл»

Новый альбом «Горизонт событий» и все хиты



DDL – data definition language (язык определения данных)

Включает всевозможные команды:

- создания (**CREATE**)
- удаления (**DROP**)
- изменения структуры (**ALTER**)

Объектов, таких, как:

- таблицы (**TABLE**)
- представления (**VIEW**)
- триггеры (**TRIGGER**)
- пользователи (**USER**)

Создание и удаление баз данных

CREATE DATABASE **[IF NOT EXISTS]** db_name

DROP DATABASE **[IF EXISTS]** db_name

Пример

```
CREATE DATABASE IF NOT EXISTS course
```





```
DROP DATABASE course
```

Кодировка базы данных и таблиц

Databases

Create database ⓘ

lab1 utf8_general_ci Create

Database	Collation	Action
<input type="checkbox"/> information_schema	utf8_general_ci	 Check privileges
<input type="checkbox"/> mysql	latin1_swedish_ci	 Check privileges
<input type="checkbox"/> performance_schema	utf8_general_ci	 Check privileges
<input type="checkbox"/> sys	utf8_general_ci	 Check privileges
Total: 4	latin1_swedish_ci	

CI case insensitive.

'ABC' = 'abc'

AS accent sensitive. 'ü'
!= 'u'

- **Кодировка** (characher set) - набор используемых СИМВОЛОВ.
- **Представление** (collation) - набор правил для сравнения символов в наборе.

Создание таблицы в GUI

Table name: Add column(s)

Structure

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index	Auto Increment
<input type="text" value="id_book"/>	<input type="text" value="INT"/>	<input type="text" value="11"/>	<input type="text" value="None"/>	<input type="text"/>	<input type="text" value="UNSIGNED"/>	<input type="checkbox"/>	<input type="text" value="PRIMARY"/>	<input checked="" type="checkbox"/>
<input type="text" value="name"/>	<input type="text" value="VARCHAR"/>	<input type="text" value="255"/>	<input type="text" value="None"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="---"/>	<input type="checkbox"/>
<input type="text" value="description"/>	<input type="text" value="TEXT"/>	<input type="text" value="1000"/>	<input type="text" value="None"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="---"/>	<input type="checkbox"/>
<input type="text" value="isbn"/>	<input type="text" value="VARCHAR"/>	<input type="text" value="35"/>	<input type="text" value="NULL"/>	<input type="text"/>	<input type="text"/>	<input checked="" type="checkbox"/>	<input type="text" value="---"/>	<input type="checkbox"/>

Table comments:

Collation:

Storage Engine:

Создание таблиц

CREATE [TEMPORARY] TABLE [IF NOT EXISTS]

tbl_name (create_definition,...)

[table_option ...]

CREATE [TEMPORARY] TABLE [IF NOT EXISTS]

tbl_name [(create_definition,...)]

[table_option ...]

select_statement

CREATE [TEMPORARY] TABLE [IF NOT EXISTS]

tbl_name

{ **LIKE** old_tbl_name | (**LIKE** old_tbl_name) }

Пример создания таблицы

```
CREATE TABLE IF NOT EXISTS `mydb`.`course` (  
  `id_course` INT NOT NULL AUTO_INCREMENT,  
  `title` VARCHAR(245) NULL ,  
  `hours` TINYINT UNSIGNED NULL,  
  PRIMARY KEY (`id_course`) )  
ENGINE = InnoDB
```

Обзор движков хранения данных MyISAM

MyISAM: Движок по умолчанию. Не поддерживает транзакций, средняя надежность хранения данных. Превосходная производительность чтения данных для интенсивных приложений.

- транзакций нет
- блокировка таблица
- полнотекстовый поиск
- поддержание целостности, внешние ключи: нет
- чувствительные к «падению» сервера, сложно восстанавливать

Обзор движков хранения данных InnoDB

InnoDB: Транзакционный тип движка, применяемый при интенсивных операциях записи. Великолепная восстанавливаемость и высокая надежность хранения данных.

- ~~• полнотекстовый индекс: нет~~
- полная поддержка транзакций (4 уровня изоляции)
- множество настроек для оптимизации
- поддержка целостности (внешние ключи)

Обзор движков хранения данных HEAP(MEMORY)

HEAP: Все в памяти. Очень быстрый поиск данных, однако все они хранятся только в памяти и будут потеряны при остановке сервера. Великолепно подходит для временных таблиц.

- транзакций нет
- репликация: да
- все данные теряются при остановке сервера (сама таблица остаётся)
- память не высвобождается при удалении записи (используется для вставки новых)

Примеры использования движков

- Финансовые транзакции — **InnoDB**
- Сессионные данные / Словари — **MyISAM**
- Локальные расчеты — **HEAP**

Удаление таблиц

DROP [TEMPORARY] TABLE [IF EXISTS]

tbl_name [, tbl_name] ...

[RESTRICT | CASCADE]

Примеры применения ALTER

```
ALTER TABLE t1 RENAME t2;
```

```
ALTER TABLE t2
```

```
MODIFY a TINYINT NOT NULL, CHANGE b c  
CHAR(20);
```

```
ALTER TABLE t2 ADD d TIMESTAMP;
```

```
ALTER TABLE t2 DROP COLUMN c;
```

```
ALTER TABLE t2 ADD c INT UNSIGNED NOT  
NULL AUTO_INCREMENT, ADD PRIMARY KEY  
(c);
```

Руководство по стилю SQL

Хороший стиль

- Идентификаторы и имена. Осмысленные и в едином стиле.
- Идентификатор начинается с `id_`. Например, `id_post` в таблице `post`
- Пробелы и отступы. Логично расставленные для лучшей читаемости кода.
- Дата и время. Соответствующие стандарту ISO 8601: `YYYY-MM-DD HH:MM:SS.SSSSS`.
- Код. Лаконичный и без излишеств, как например: ненужные кавычки или скобки.
- Комментарии. Предпочтительно в стиле C — `/*` (начало) и `*/` (конец). Либо `--` перед комментарием, тогда окончанием будет новая строка.

Руководство по стилю SQL

Плохой стиль

- Приписка к атрибутам названия таблицы.
- Префиксы и венгерская нотация. Префиксы наподобие `sp_` или `tbl_` избыточны.
- Множественное число. Лучше использовать более естественно звучащие собирательные понятия. Например, `staff` вместо `employees` или `people` вместо `individuals`.
- Идентификаторы в кавычках. Если они обязательно нужны, тогда используйте двойные кавычки, определённые в стандарте SQL-92 с целью лучшей переносимости в дальнейшем.