Основы баз данных

1. Целостность значений в ячейках баз данных является основой MySql
   1. Удаление строки в род таблице ведёт к каскадному удалению строки в дочерней таблице(cascade)
2. Нормальные формы:

2.1) Первая нормальная форма (Значение в одной ячейке атомарно, т.е. его не надо парсить на несколько значений, если это не так-то надо разбить одно поле на несколько)

2.2) Вторая нормальная форма (Значения уже в первой нормальной форме и значения столбцов полностью зависят от значения первичного (составного или несоставного) ключа, если это не так, то столбцы переносим в другую таблицу, чтобы выполнялась вторая нормальная форма.

2.3) Третья нормальная форма (Значения уже во второй нормальной форме и любой не ключевой атрибут зависит только от первичного ключа)

Работа с таблицами database

1. Create database FirstDB; // Создать бд
2. drop database firstdb; // Удалить бд
3. use FirstDB; // Использовать бд
4. show databases; // Показать все доступные бд

Создание таблицы

CREATE TABLE Orders

(id SERIAL,

date DATE,

product\_id BIGINT UNSIGNED NOT NULL,

qty INT UNSIGNED,

amount DECIMAL(10,2),

customer\_id BIGINT UNSIGNED,

PRIMARY KEY (id),

FOREIGN KEY (product\_id) REFERENCES Products (id)

ON DELETE RESTRICT ON UPDATE CASCADE,

FOREIGN KEY (customer\_id) REFERENCES Customers (id)

ON DELETE RESTRICT ON UPDATE CASCADE);

Изменение структуры таблиц

Show tables // Посмотреть информацию о бд

describe customers; // Узнать информацию о таблице

show create table customers; // Узнать информацию о таблице

alter table customers add new\_column varchar(20) after address;

// Добавили новый столбец в таблицу

alter table customers change zhopa opa text;

// Изменили название столбца и его тип

alter table customers drop name;

//Удалить столбец

alter table customers DROP PRIMARY KEY;

// Удаляем первичный ключ

Добавление строк в таблицу

insert into customers (phone, address)

values("dd", "32332"),

("ee", "ew2");

Извлечение данных из таблиц

select \* from customers;

select distinct ratings from customers; // выводит значения без повторов

//// Сортируем по убыванию

select distinct \* from money

order by money desc, firsname;

// делаем выборку с условием

select \* from money

where money > 130

order by id ;

Обновление поля таблицы

1) update billing set currency = "temp"

where sum > 200;

Удаление поля таблицы

1. delete from billing

where payer\_email is Null or recipient\_email is Null or payer\_email = '' or recipient\_email = '';

Функции агрегации

1. select count(1) from project; // Кол-во по столбцу
2. select avg(budget) from project; // Среднее значение по столбцу
3. select

project\_finish,

project\_start,

datediff(project\_finish, project\_start)

from project

where project\_finish is not null; // datediff() выводит разницу между двумя датами

4)

select

min(datediff(project\_finish, project\_start)),

max(datediff(project\_finish, project\_start))

from project

where datediff(project\_finish, project\_start) >= 0;

// Значения min, max;

5)

select

max(datediff(project\_finish, project\_start)),

client\_name

from project

where project\_finish >= project\_start

group by(client\_name);

// Значения group by(), нужно, так как если использовать агрегированную функцию она вернёт одно значение, а при group by(). Значения для группы в скобках.

6)

select

client\_name,

project\_name,

max(datediff(project\_finish, project\_start)),

min(datediff(project\_finish, project\_start))

from project

where project\_finish >= project\_start

group by(client\_name)

order by (project\_start)

Limit 20;

(Полный скрипт с лимитами Limit 20 на количество выводимых позиций).

Модуль 2 Основы реляционных баз данных

1. select \* from category , product; //Получили Декартово произведение
2. select product.product\_name, product.price, c.category\_name, count(\*)

from product inner join category as c on product.category\_id = c.category\_id

group by (product\_name);

// Использование inner join

1. select good.name as 'good\_name', category.name as 'category\_name '

from

good inner join category\_has\_good

on good.id = category\_has\_good.good\_id

inner join category

on category.id = category\_has\_good.category\_id

order by good.name, category.name;

// Использование нескольких inner join подряд

1. select \* from product as p

right outer join category as c

on p.category\_id = c.category\_id;

// right outer join

1. select \* from product as p

left outer join category as c

on p.category\_id = c.category\_id;

// left outer join

1. select \* from product where price > 900

union

select \* from product where price < 300

// Union

7)

SELECT good.name as good\_name, category.name as category\_name FROM good

LEFT OUTER JOIN category\_has\_good on category\_has\_good.good\_id = good.id

LEFT OUTER JOIN category on category.id = category\_has\_good.category\_id

UNION

SELECT good.name as good\_name, category.name as category\_name FROM good

RIGHT OUTER JOIN category\_has\_good on category\_has\_good.good\_id = good.id

RIGHT OUTER JOIN category on category.id = category\_has\_good.category\_id

ORDER BY good\_name, category\_name

// Реализация full join в Mysql