**POINTS ABOUT WORK WITH DATABASE**

PreparedStatement (подготовленные запросы)

<http://articles.javatalks.ru/articles/2>

Начнем с того, что СУБД разбирает каждый запрос перед выполнением, оптимизирует и создает "план" =) (query plan). Если один и тот же запрос выполнять несколько раз, СУБД его кеширует, точнее тот самый план, и больше не разбирает и не оптимизирует. Благодаря этому он выполняется быстрее.  
Разница между Statement и PreparedStatement в шапке темы в том, что

1. INSERT INTO tb\_1 (name,fname,tel,adres) VALUES (?,?,?,?)

- это один и тот же запрос (будет закеширован), а

1. INSERT INTO tb\_1 (name,fname,tel,adres) VALUES ('" + name1 + "','" + fname1 + "','" + tel1 + "','" + adres1 + "')

- каждый раз новый запрос, т.к. параметры подставляются непосредственно, и, соответственно, кешироваться не будет.

Если запрос выполняется часто и особенно с большим количеством условий, то лучше использовать PreparedStatement, в таком случае производится прекомпиляция и оптимизация запроса один раз, а не при каждом вызове.  
Если запрос используется не часто или простой, то можно использовать обычный стейтмент, чтобы не забивать пул прекомпилированных стейтментов. Либо делать выбор на основании удобства использования

PreparedStatement является предкомпилированным, то он может быть эффективно использован множество раз. PreparedStatement считается лучшим выбором нежели Statement, т.к. он автоматически обрабатывает специальные символы, а так же предотвращает, так называемые, SQL injection attack (когда в запрос можно подставить свой код).

#### Какие преимущества в использовании PreparedStatement над Statement?

* PreparedStatement позволяет предотвратить атаки типа SQL injection, т.к. он автоматически экранирует специальные символы.
* PreparedStatement позволяет использовать динамические запросы с внедрением параметров.
* PreparedStatement быстрее Statement. Это особенно заметно при частом использовании PreparedStatement или при использовании для вызова группы запросов.
* PreparedStatement позволяет писать объектно ориентированный код с использованием сеттеров\геттеров. В то время при использовании Statement необходимо использовать конкатенацию строк для создания запроса. Для больших запросов конкатенация выглядит, как минимум, большой, а так же несет в себе большой риск ошибки в запросе.

# **14 вопросов об индексах в SQL Server, которые вы стеснялись задать**

**https://habrahabr.ru/post/247373/**

**Индексы Design SQL Server 2000**

<http://www.sql.ru/articles/mssql/03013101indexes.shtml>

## [Индексы в MySQL](https://ruhighload.com/post/%D0%A0%D0%B0%D0%B1%D0%BE%D1%82%D0%B0+%D1%81+%D0%B8%D0%BD%D0%B4%D0%B5%D0%BA%D1%81%D0%B0%D0%BC%D0%B8+%D0%B2+MySQL)

## <https://ruhighload.com/%D0%B8%D0%BD%D0%B4%D0%B5%D0%BA%D1%81%D1%8B>

# Индексы в MySQL

<https://ruhighload.com/post/%D0%A0%D0%B0%D0%B1%D0%BE%D1%82%D0%B0+%D1%81+%D0%B8%D0%BD%D0%B4%D0%B5%D0%BA%D1%81%D0%B0%D0%BC%D0%B8+%D0%B2+MySQL>

# Индексы на примере Mysql

<https://ruhighload.com/index.php/2009/06/09/%D0%BA%D0%B0%D0%BA-%D0%B2%D1%8B%D0%B1%D1%80%D0%B0%D1%82%D1%8C-%D0%BA%D0%BE%D0%BB%D0%BE%D0%BD%D0%BA%D0%B8-%D0%B4%D0%BB%D1%8F-%D0%B8%D0%BD%D0%B4%D0%B5%D0%BA%D1%81%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD/>

# Профилирование в MySQL

<https://ruhighload.com/index.php/2009/07/28/maatkit-%D1%80%D0%B0%D1%81%D1%88%D0%B8%D1%80%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5-%D0%B2%D0%BE%D0%B7%D0%BC%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D0%B8-mysql/>

# EXPLAIN в Mysql

<https://ruhighload.com/post/EXPLAIN+%D0%B2+Mysql>

# Как использовать SHOW PROFILE в MySQL?

<https://ruhighload.com/post/%D0%9A%D0%B0%D0%BA+%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D1%8C+SHOW+PROFILE+%D0%B2+MySQL%3F>

# Оптимальная настройка Mysql

<https://ruhighload.com/index.php/2009/04/23/%D0%BE%D0%BF%D1%82%D0%B8%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F-%D0%BD%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B9%D0%BA%D0%B0-mysql-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80%D0%B0/>

# Как включить slow log в MySQL?

<https://ruhighload.com/post/%D0%9A%D0%B0%D0%BA+%D0%B2%D0%BA%D0%BB%D1%8E%D1%87%D0%B8%D1%82%D1%8C+%D0%BB%D0%BE%D0%B3+%D0%BC%D0%B5%D0%B4%D0%BB%D0%B5%D0%BD%D0%BD%D1%8B%D1%85+%D0%B7%D0%B0%D0%BF%D1%80%D0%BE%D1%81%D0%BE%D0%B2+%D0%B2+MySQL%3F>