php и nodejs, разница на пальцах

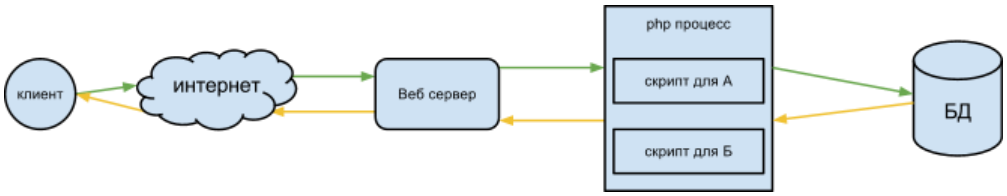
Являясь постоянным пользователем форума nodejs.ru, часто наблюдаю картину когда люди начиная изучать nodejs сравнивают ее с php, а иногда пытаются работать с ней так как с php. Я бы хотел объяснить “на пальцах” разницу между php и nodejs применительно к работе сайта. Статья предназначена для новичков. Я намеренно буду говорить очень упрощенно, не вдаваясь в глубокие подробности, что бы как можно проще показать различия в технологиях.

Что то объяснять всегда лучше на наглядном примере с картинками. Поэтому придумаем небольшой “сферический сайт в вакууме” и примем некоторые условия.

Пусть у нас имеется некий сайт, который понимает всего два запроса:
Запрос А выполняется за 1 секунду, он не требует обращение к БД.
Запрос Б выполняется за 5 секунд, причем 4 из них, он тратит на ожидание ответа БД.
Так же условимся что время между запросами не менее 1 секунды.

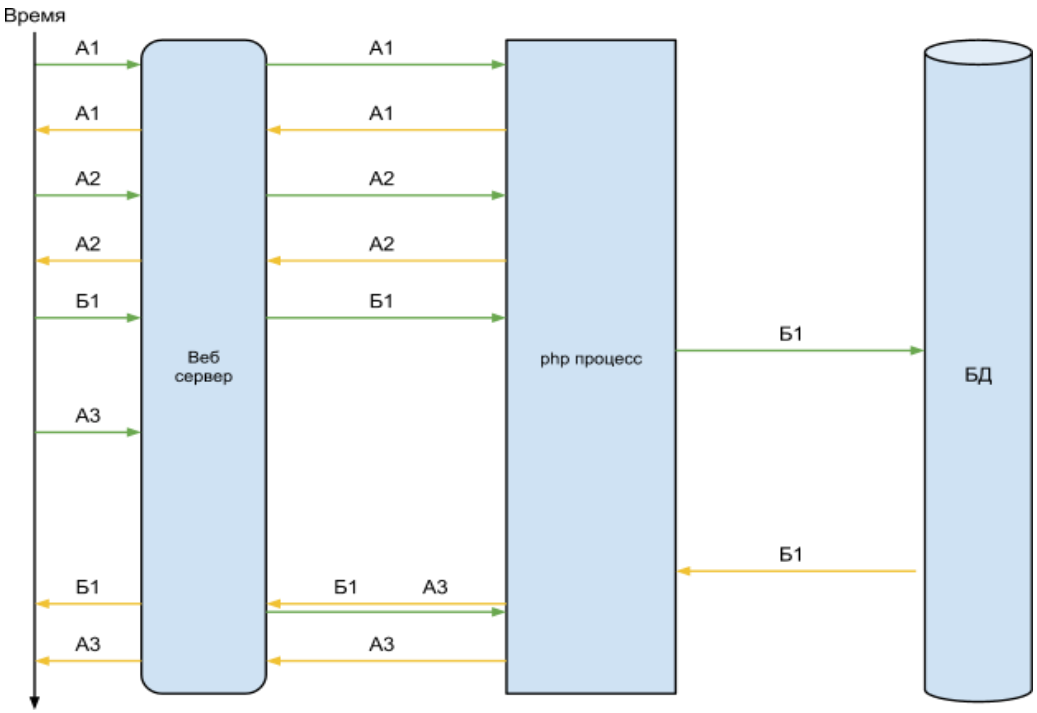
PHP

Давайте рассмотрим как это работает на php.
В самой упрощенной форме архитектура сервера выглядит так:



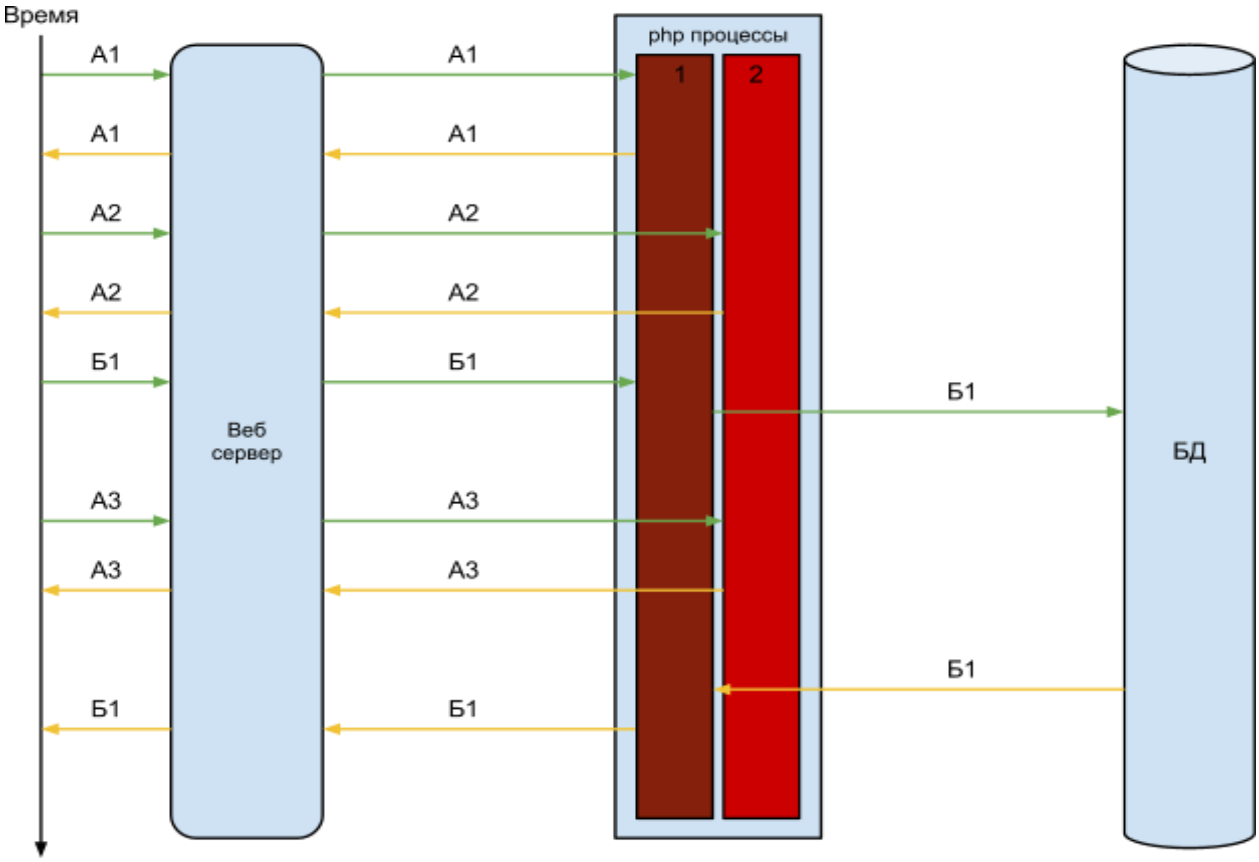
Важно тут следующие, веб сервер получив запрос от клиента передает его в php процесс. В свою очередь процесс php в один момент времени может обрабатывать один запрос, по завершению работы, результат возвращается веб серверу, а сам процесс перестает существовать. Веб сервер получая ответ отправляет результат клиенту и закрывает соединение.

При наличии всего одного php процесса, работа нашего сервера можно отобразить на такой схеме:



Из схемы видно, что пока к нам поступают только запросы А наш сервер бодро на них отвечает и в целом выполняет поставленным задачи, но как только к нам приходит запрос Б, сервер перестает отвечать на запросы, до момента пока не будет готов ответ на запрос Б. Так же на схеме видно что большую часть времени запроса Б “все” ждут результат работы базы данных.

Для решения этой проблемы приходиться увеличивать кол-во php процессов, давайте увеличим до 2х, в результате схема принимает такой вид:



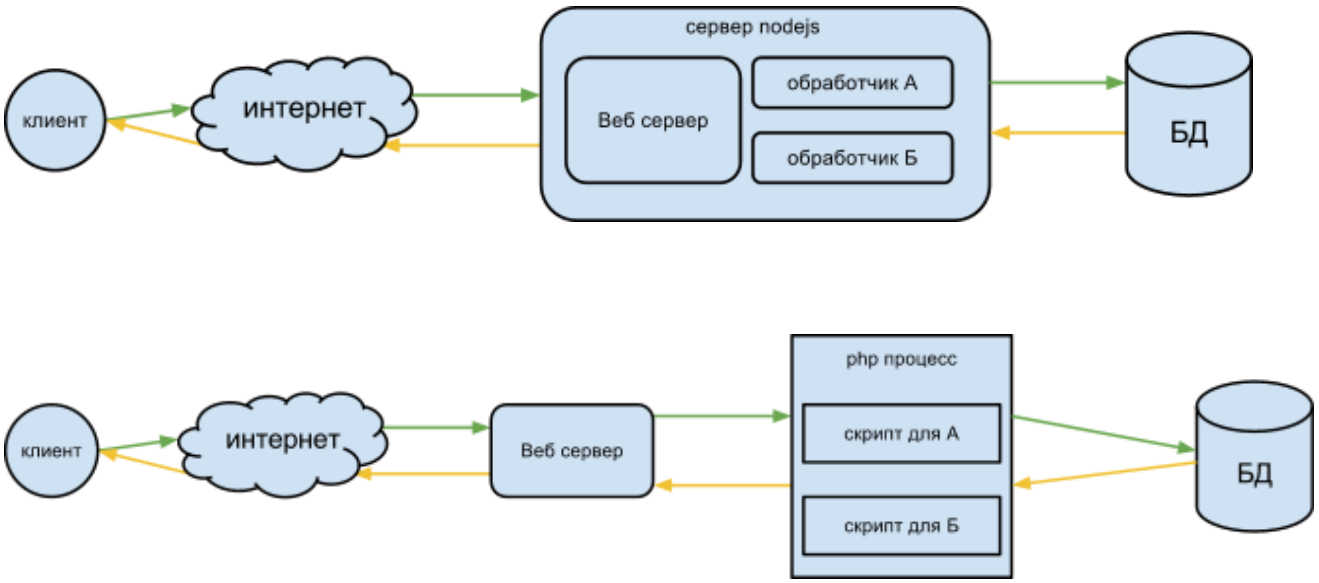
Из этой схемы видно, что запрос Б “повисает” в обработке в первом php процессе, при этом сервер продолжает отвечать на остальные запросы. Все будет идти хорошо до момента, когда к нам не придут два запроса Б, тогда оба php процесса “повиснут” в ожидании ответа базы, и сервер в целом перестанет отвечать, до момента пока один из них не освободиться.

Ну мы то уже знаем что делать? Правильно, возьмем и увеличим кол-во php процессов, сразу до 20 или 30 и проблема вроде как ушла, хотя на самом деле проблема просто немного отдалилась и момент когда придет 30 запросов Б наступит. Вся беда в том что мы не можем создавать бесконечно много php процессов и путь наращивать их в запредельных количествах неверен.

Самое главное в что следует вынести из этих схем, это то, что операции работы с базой данных в php выполняются синхронно. В нашем случае процесс выполнивший запрос к базе неспособен обработать другие запросы и вынужден “висеть” (ничего не делая) ожидая ответ от базы данных.

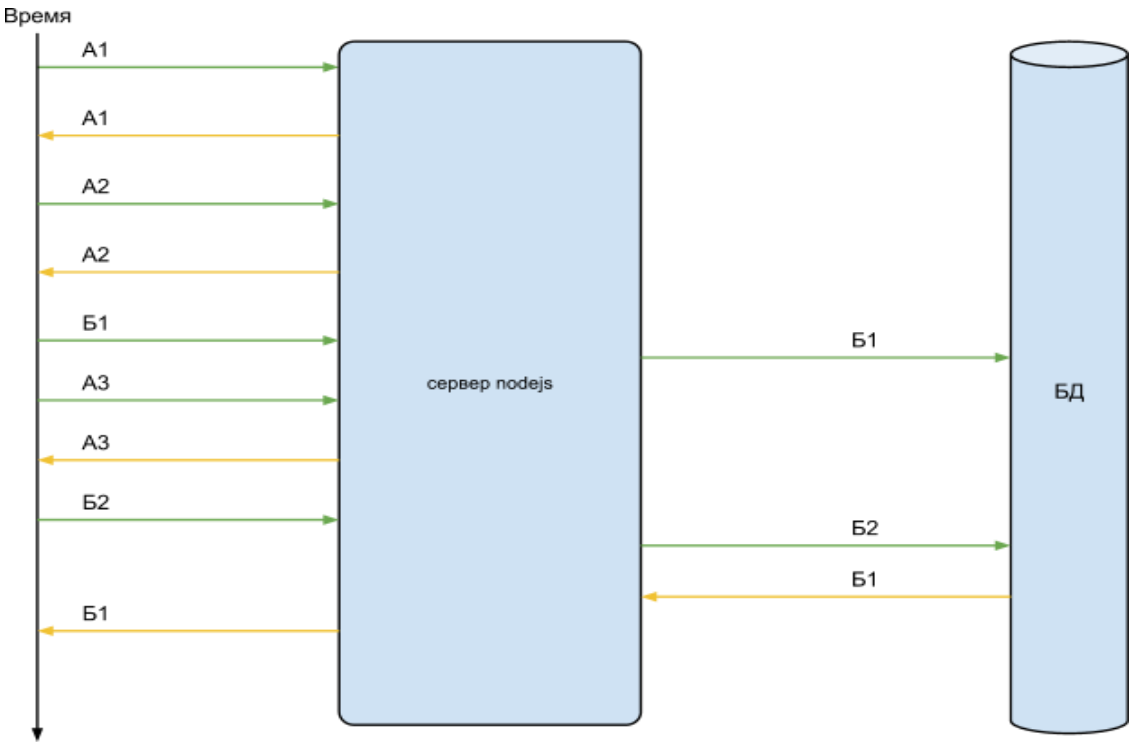
nodejs

Что дает нам nodejs?
Сначала посмотрим как выглядит простой сервер:



Сразу бросается в глаза то, что сервер включает в себя обработчики непосредственно запросов А и Б, а так же сам Веб сервер. Всё это добро крутится в одном node процессе и постоянно висит в памяти.

Посмотрим на схему работы:



На схеме хорошо видно что запросы Б не приводят к “подвисанию” сервера в ожидании ответа базы. Сервер получив запрос Б, просто сформирует и отправит запрос в базу данных, и продолжит отвечать на остальные запросы, как только ответ от базы будет получен, сервер вернет результат клиенту. В случае nodejs неважно как и в каком количестве придут запросы Б, ни один из них не приведет к “подвисанию” в ожидании ответа базы.

Вывод

А вывод прост.
Придя в nodejs не пытайтесь делать что то так, как вы делали это в php.
Помните что вы работает в асинхронной среде, не используйте операции приводящие к блокировке, этим вы убиваете идею nodejs.
Помните что nodejs обрабатывает множество запросов в одном процессе постоянно висящем в памяти, поэтому следите за своими переменными и тем как вы расходуете память.
Не стоит запускать 50 процессов node для одного сайта, да и вообще не стоит запускать их больше кол-ва ядер на процессоре, большее их количество только замедлит работу в целом.
nodejs это не серебряная пуля и одно лишь ее использование не решит проблем масштабирования и работой под большой нагрузкой.