*УДК 004.652.5*

А.Ю. Пыркова, М.Б. Борисенко

*Казахский национальный университет им. аль-Фараби, Алматы, Казахстан*
*E-mail: Anna.Pyrkova@kaznu.kz, Mstislav.Borissenko@gmail.com*

# Перспективы использования объектных баз данных для хранения и защиты информации, используемой в Web-приложениях

В данной работе представлены основные принципы разработки защищенн ых веб-приложений, технологии, которые могут быть при этом использованы, а та кже типы атак, против которых должна быть предоставлена защита. При этом были рассмотрены перспективы использования объектных баз данных для хранения инфо рмации, используемой в защищенных веб-приложениях, проанализированы сильные и слабые стороны объектно-ориентированных систем управления базами данных, а также возможности по защите информации на примере реализации Intersystems Ca ché. В работе также было рассмотрено, почему в современном мире популярны реляцион ные базы данных, какие их особенности и возможности актуальны и необходимы, р ассмотрено, какие из требуемых характеристик обеспечиваются также и объектными сист емами управления базами данных. Рассмотрены протоколы безопасности и шифро вания, а также типы ключей, поддерживаемые коммерческими продуктами. В раб оте представлено, почему производительность систем управления базами данных ва жна при их использовании для хранения информации, используемой в веб-прил ожениях. Были рассмотрены различные способы доступа к Cache Database и проведен срав нительный анализ их производительности при доступе к данным, хранимым в базе, из программы, написанной на языке программирования Java. На основе полученных данных было установлено, что объектные базы данных перспективны с точки зрения использования в реальных современных приложениях, а также определено, к акой из способов доступа к данным предпочтителен в различных ситуациях.

***Ключевые слова***: объектно-ориентированные базы данных, производительность баз данных, веб-приложения, защита информации.

*A.Yu. Pyrkova, M.B. Borissenko*

## Perspectives of object databases for storage and protection of information used in Web-applications

Main principles of protected web-applications development, technolog ies that may be used and types of attacks, against which defence must be provided, are pr esented in this work. Perspectives of object databases use for storing and protecting information that is manipulated from protected web-applications are considered, stren gths and drawbacks of object-oriented database management systems as well as capabiliti es for information protection are analyzed on the example of Intersystems Caché. Reasons of relational database management systems' popularity in the modern world are also considered in this work, as well as actuality and necessity of several their features and cap abilities, and which of the necessary characteristics are supported by object database man egement systems as well. Security and encryption protocols are considered in this article, as well as types of keys that are supported by commercial products. Importance of datab ase management

systems' performance is explained for the situation when they are used to store information processed by web-applications. Different data access types are considered and their comparative analysis is provided using the example of Java application program accessing information stored in Cache Database. On the base of acquired data, it was found out that object-oriented databases are perspective for use in real modern applications. It was also determined, which data access types are preferable in different situations.
***Key words***: object-oriented databases, database performance, web-applications, data protection.

*А.Ю. Пыркова, М.Б. Борисенко*

**Web-қосымшаларда қолданылатын ақпаратты сақтау және қорғау үшін объект дерекқорлардың қолдануының перспективалары**

Бұл жұмыста негізгі қорғалған веб-қосымшарадың қағидаттары және қолдануға болатын технологиялар көрсетілген, шабуылдар қарсы қорғаныш керектігі туралы айтылған. Веб-қосымшаларда қолданылатын ақпаратты сақтау және қорғау үшін объект дерекқорлардың қолдануының перспективалары, объект дерекқорлар басқару жүйелердін күшті және әлсіз жақтары, ақпарат қорғауға мүмкіншіліктер Intersystems Caché мысалмен талдалған. Бұл жұмыста реляциялық дерекқорлардың бүгінгі әлемде әйгілігі түсіндірілген, олардың маңызды және керекті ерекшіліктері көрсетілген, олардан қандай ренедіктер объект дерекқорлардың басқару жүйелер де қамтамасыз етеді. Қауіпсіздік және мұқамдау протоколдар және коммерциялық өнімдерде қолданылатын кілттер зерттелген. Жұмыста веб-қосымшаларда қолданылатын ақпаратты сақтау үшін арналған дерекқорлардың басқару жүйелердің өнімділігінің маңыздылығы көрсетілген. Әр — түрлі Cache Database-де сақталатын ақпаратқа рұқсат әдістері қараған, Java бағдарламалау тілде жазған қосымшада олардың өнімділігінің салыстырмалы талдауын жүргізген. Алынған нәтижесінің негізінде объект дерекқорлардың нақты бүгінгі қосымшаларда қолдану перспективалары және әр — түрлі жағдайда ұнамды рұқсат әдістері анықталған.
***Түйін сөздер***: объект дерекқорлар, дерекқорлардың өнімділігі, веб-қосымшалар, ақпарат қорғауы.

## Актуальность работы в современном мире

В современном мире всё большее распространение получают веб-приложения. Они стали использоваться во всех отраслях современного бизнеса и компаниями всех масштабов, от малого бизнеса до международных корпораций. Тем не менее, многие из создаваемых приложений разрабатываются только с необходимой базовой функциональностью, требуемой компанией-заказчиком, без использования каких-либо технологий повышения безопасности. В результате этого создаваемые приложения легко поддаются взлому и могут быть использованы для удаленного проникновения на компьютеры и серверы компании с различными целями, начиная от просмотра, изменения и хищения информации, и заканчивая исполнением произвольного программного кода.

Практически в любом веб-приложении для хранения используемой сервером информации используются базы данных. Это означает, что не только само приложение может быть целью атаки злоумышленников, но и используемая приложением база данных. Поскольку система защищена так же хорошо, как самый незащищенный из ее компонентов, необходимо разрабатывать систему защиты не только для самого приложения,

но и для базы данных, в которой хранится информация. Соответственно, используемая система управления базами данных должна предоставлять широкие возможности по обеспечению безопасности.

В современных приложениях наиболее часто используются реляционные базы данных, поскольку они максимально приспособлены для работы с информацией, легко представимой в табличной форме. Реляционные СУБД хорошо зарекомендовали себя на рынке и предоставляют широкие возможности как по работе с данными, так и по их защите. Например, одна из самых защищенных систем, Oracle Database, предоставляет все необходимые функции, включая ограничение привилегий пользователей, автоматическую блокировку при попытке взлома, блокировку дополнительных sql-команд после разделителя строк, хранимые процедуры и триггеры, которые могут быть использованы для обеспечения безопасности, возможности по шифрованию информации и ограничению доступа. Это означает, что любая альтернатива реляционным СУБД должна иметь соответствующие возможности и не уступать в функционале, чтобы быть конкурентоспособной.

Поскольку используемые данные становятся всё более сложными и взаимосвязанными, а разрабатываемые приложения всё в большем числе создаются с использованием объектно-ориентированного подхода, представляется интересным использовать объектные базы данных вместо реляционных.

**Основные результаты**

Объектные системы управления базами данных представлены гораздо меньшим числом продуктов, чем реляционные, и впервые появились намного позже. Одной из наиболее известных коммерческих объектных СУБД является Caché Database от компании Intersystems. Она широко применяется в системе здравоохранения многих стран, включая США, в крупных компаниях, связанных с производством и строительством. Cache является интересным примером объектных СУБД, поскольку предоставляет возможность использования не только объектного доступа к хранящимся данным, но и реляционного sql-доступа, а также прямого доступа к данным. Это реализовано при помощи использования двух словарей данных, в результате чего на основе каждого класса автоматически формируется таблица, а каждая таблица представляется как класс [1]. В то же время, она соответствует всем предъявляемым к объектным СУБД требованиям, включая поддержку сложных и составных объектов, множеств, списков и массивов, классов и типов, а также их иерархии, инкапсуляции, наследования и полиморфизма [2]. При этом, как и любая СУБД, Caché обеспечивает расширяемость, стабильность и возможность восстановления после аппаратных и программных сбоев.

Создание базы данных в Cache значительно отличается от создания реляционной базы данных [3,4]. При создании новых классов объектов необходимо провести компиляцию созданных классов. Создаваемые классы могут иметь свойства, являющиеся аналогам столбцов таблиц в реляционных СУБД; методы, являющиеся аналогом хранимых процедур; могут получить значения параметров, которые будут использоваться по умолчанию. Можно прикрепить к классу триггер таким же образом, как и в реляционных СУБД: он может срабатывать при добавлении, удалении или изменении данных, а также до или после того, как операция будет произведена. Таким образом, предоставляется функционал по работе с данными, схожий с функционалом реляционных

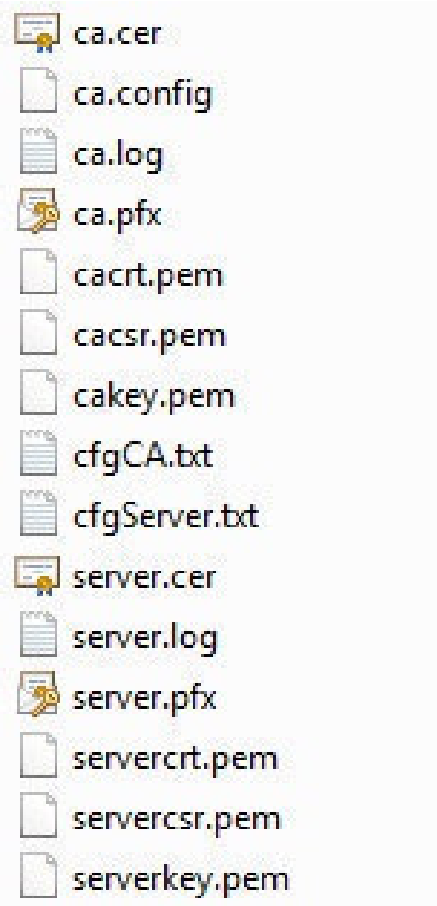


**Рисунок 1.** Выпущенные сертификаты

СУБД.

Как и большинство коммерческих систем управления базами данных, Cache позволяет создавать пользователей, указывать их роли, управлять их привилегиями, запрещать или разрешать доступ к определенным таблицам, представлениям и процедурам [5]. Настройка привилегий позволяет запрещать или разрешать изменение структуры базы данных - создание, удалении или изменение таблиц, триггеров, хранимых запросов, представлений, функций и процедур. Отдельно для каждой таблицы могут быть предоставлены права на выборку, вставку, изменение и удаление данных. Таким образом, предусмотрена возможность настраивать привилегии пользователей так же, как и в реляционных СУБД.

Кроме того, Caché предоставляет возможности по шифрованию хранимых данных [6]. Для этого необходимо использовать ключ шифрации базы данных. Также поддерживается настройка SSL/TLS конфигурации, позволяющей посылать запросы к базе данных в зашифрованном виде. Поддерживаются протоколы TLS первой версии, а также SSL второй и третьей версий. Они используют сертификаты X.509 и, соответственно, асимметричное шифрование для аутентификации противоположной стороны, а также для обмена симметричным ключом. Также эти протоколы позволяют использовать аутентификацию подключающегося клиента, либо отказаться от нее. Cache поддерживает секретные ключи типов RSA и DSA. Таким образом, возможно не только зашифро-

вать данные, хранящиеся в базе, но и использовать защищенный канал передачи данных при использовании удаленного доступа.

Несмотря на то, что в коммерческих приложениях необходимо использовать только сертификаты, выданные зарегистрированным центром сертификации, для обеспечения возможности проверки их подлинности, в тестовых целях и при начальной разработке приложений возможно использовать самостоятельно изданные сертификаты. Для этого была использована свободно распространяемая программа OpenSSL, с помощью которой сначала был выпущен сертификат X.509 собственного центра сертификации, на основе которого затем были выпущены сертификаты для сервера и клиента, каждый со своими секретными ключами. Во всех случаях были использованы ключи RSA. В случае если аутентификация клиента не требуется, клиентский сертификат создавать и использовать не требуется. Выпущенные сертификаты затем были использованы для установки тестового соединения, которое было успешно установлено.

Тем не менее, даже в случае использования защищенного соединения необходимо обеспечить защиту от SQL-инъекций, поскольку атака на базу данных может быть проведена либо аутентифицированным пользователем, либо злоумышленником с аутентифицированной машины без ведома удостоверившего свою личность пользователя. Подобная защита может быть реализована экранированием специальных символов, фильтрацией поступающего в базу данных запроса, либо комбинацией этих способов, что и было произведено. Фильтрация поступающей строки происходила на предмет наличия запрещенных специальных символов, таких как точка с запятой, которых не может быть в стандартном запросе, а также на запрещенные буквосочетания, соответствующие командам SQL.

Альтернатива распространенным реляционным СУБД должна не только обеспечивать похожие возможности, чтобы быть конкурентоспособной, но и обладать положительными особенностями, выгодно отличающими ее от используемых систем. Как одно из основных преимуществ можно рассматривать более высокий уровень абстракции при работе с данными, поскольку вся хранимая в объектной базе информация представляется в виде объекты. Это позволяет использовать хранимые объекты в программах точно так же, как и любые другие объекты языка программирования, на котором пишется приложение. Caché позволяет автоматически создавать проекции своих классов в виде классов Java, .NET и C++. Таким образом, разрабатывать приложения с использованием объектных баз данных может быть проще специалистам, незнакомым с языком запросов SQL, но успешно использующим объектно-ориентированный подход.

Во многих случаях, особенно в системах реального времени, критически важной характеристикой приложения является быстрота его отклика. Соответственно, очень важно, чтобы используемая СУБД позволяла быстро производить большое количество операций с данными, запросы на выполнения которых могут поступать от различных пользователей. Поскольку Caché поддерживает различные способы доступа к информации, хранящейся в базе данных, представляет интерес сравнение производительности различных методов. Были проведены тесты производительности трех способов доступа к данным: реляционного доступа с использованием драйвера JDBC, реляционного доступа с использованием источника данных ODBC, а также объектного доступа с использованием драйвера JDBC. Тесты проводились на наборе 10 тысяч, 100 тысяч и 10 миллионов операций выборки данных Select из одной таблицы. Результаты показаны в

Таблице 1.

Таблица 1. Сравнение производительности различных способов доступа к данным

| **Тип доступа к данным** | **$10^4$ операций** | **$10^5$ операций** | **$10^7$ операций** |
|---|---|---|---|
| Объектный доступ | 1800 мс | 15 с | 21 мин 33 с |
| SQL-доступ ODBC | 1450 мс | 11.5 с | 19 мин 14 с |
| SQL-доступ JDBC | 790 мс | 7.8 с | 12 мин 51 с |

Во всех случаях производительность реляционного доступа оказалась существенно выше. Самым эффективным способом доступа к данным на рассмотренных примерах оказался вариант с использованием драйвера JDBC. Доступ с использованием источника данных ODBC происходил в полтора, а объектный доступ - почти в два раза медленнее, чем показавший наилучшую производительность доступ с использованием драйвера JDBC. Таким образом, плата за повышение уровня абстракции в этом случае оказалась достаточно высокой.

Это означает, что в случаях, когда производительность и скорость выполнения запросов к базе данных критически важна, объектный доступ к данным использовать нежелательно, что уменьшает привлекательность использования объектных баз данных, поскольку использование табличных отображений классов никак не отличается от работы с информацией, хранящейся в реляционных базах данных.

Таким образом, объектные базы данных в целом и Caché Database в частности предоставляют весь необходимый функционал как по работе с данными, так и по их защите. При этом все коммерческие базы данных обладают своими сильными и слабыми сторонами, поэтому для каждого приложения необходимо выбирать СУБД, максимально подходящую в данном конкретном случае. Объектные базы данных отлично подходят для использования в объектно-ориентированных приложениях, написанных на Java и других языках программирования, но при этом необходимо оценить, какое количество пользователей сможет использовать приложение без снижения производительности. В некоторых случаях использование объектного доступа к данным может быть нежелательно из-за его более медленного выполнения, и в этих случаях использование объектных баз данных может быть эквивалентно использованию реляционных.

## Литература

[1] *Финн М.* Сравнение производительности в реальных условиях: http://intersystems.ru/cache/whitepapers/pdf/cache-vs-rdbms.pdf

[2] *Atkinson M., Bancilhon F., DeWitt D., Dittrich K., Maier D., Zdonik S.* The Object-Oriented Database System Manifesto. – New York: Elsevier Science, 1990. – 17 p.

[3] *Харрингтон Д.* Проектирование объектно-ориентированных баз данных. – М: ДМК Пресс, 2012. – 272 с.

[4] *Джордан Д.* Обработка объектных баз данных в C++. Программирование по стандарту ODMG. – М: Вильямс, 2001. – 384 с.

[5] *Кирстен В., Иринге р М., Кюн М., Рериг Б.* Постреляционная СУБД Cache' 5. Объектно-ориентированная разработка приложений. – Москва : Бином-Пресс, 2003. – 402 с.

[6] *Труб И.И.* СУБД Cache: работа с объектами. – М: Диалог-МИФИ, 2006. – 480 с.

## References

[1] *Finn M.* Sravneniye proizvoditel'nosti v real'nykh usloviyakh: http://intersystems.ru/cache/whitepapers/pdf/cache-vs-rdbms.pdf

[2] *Atkinson M., Bancilhon F., DeWitt D., Dittrich K., Maier D., Zdonik S.* The Object-Oriented Database System Manifesto. – New York: Elsevier Science, 1990. – 17 p.

[3] *Kharrington D.* Proyektirovaniye obyektno-oriyentirovannykh baz dannykh. – Moskva: DMK Press, 2012. – 272 s.

[4] *Dzhordan D.* Obrabotka obyektnykh baz dannykh v C++. Programmirovaniye po standartu ODMG. – M: Vil'yams, 2001. – 384 s.

[5] *Kirsten V., Iringer M., Kyun M., Rerig B.* Postrelyatsionnaya SUBD Cache' 5. Obyektno-oriyentirovannaya razrabotka prilozheniy. – M: Binom-Press, 2003. – 402 s.

[6] *Trub I.I.* SUBD Cache: rabota s obyektami. – M: Dialog-MIFI, 2006. – 480 s.