**InterSystems Caché**

1. **История развития СУБД**

InterSystems Caché - это коммерческая операционная система управления базами данных от InterSystems, используемая для разработки программных приложений для управления здравоохранением, банковского дела и финансовые услуги, правительство и другие секторы. Программное обеспечение заказчика может использовать базу данных с объектом и кодом SQL. Caché также позволяет разработчикам напрямую управлять своими базовыми структурами данных: иерархическими массивами, известными как MUMPS технология.

Официальная дата выхода СУБД Caché компании InterSystems — 1997 г. Однако ее история восходит к началу 70-х и связана с легендарными компьютерами PDP компании DEC. Изначально СУБД Caché создавалась как универсальная система программирования для Массачусетского госпиталя. Технология оказалась удачной и постепенно стала проникать и в другие отрасли; сейчас Caché активно используется в промышленности, финансах, торговле и т. д.

1. **Инструменты для взаимодействия с СУБД**

Cache поддерживает как SQL, так и объектный доступ.

СУБД Cache - это открытая система; она предоставляет огромное количество интерфейсов для взаимодействия с внешним миром. Администрирование Cache осуществляется двумя средствами : Configuration Manager, Control Panel.

InterSystems Caché поддерживает платформы Microsoft Windows, UNIX, Linux, OpenVMS, Mac OS X. Caché предоставляет различные интерфейсы к .NET — SQL, объектный, SOAP — и обеспечивает целостное взаимодействие с платформой .NET Framework. Основной механизм, предоставляющий доступ к Caché из .NET-приложений, — это Caché Managed Provider для .NET.  Cache Object Server (Сервер объектов Cache) позволяет использовать объекты Cache в Java, ActiveX и C++, а также во многих других средствах разработки приложений. Cache SQL Server (Реляционный сервер Cache) обеспечивает высокопроизводительный доступ к базам данных Cache через SQL и ODBC. Cache Поддерживает общие интернет-протоколы: POP3, SMTP, MIME, FTP и другие.

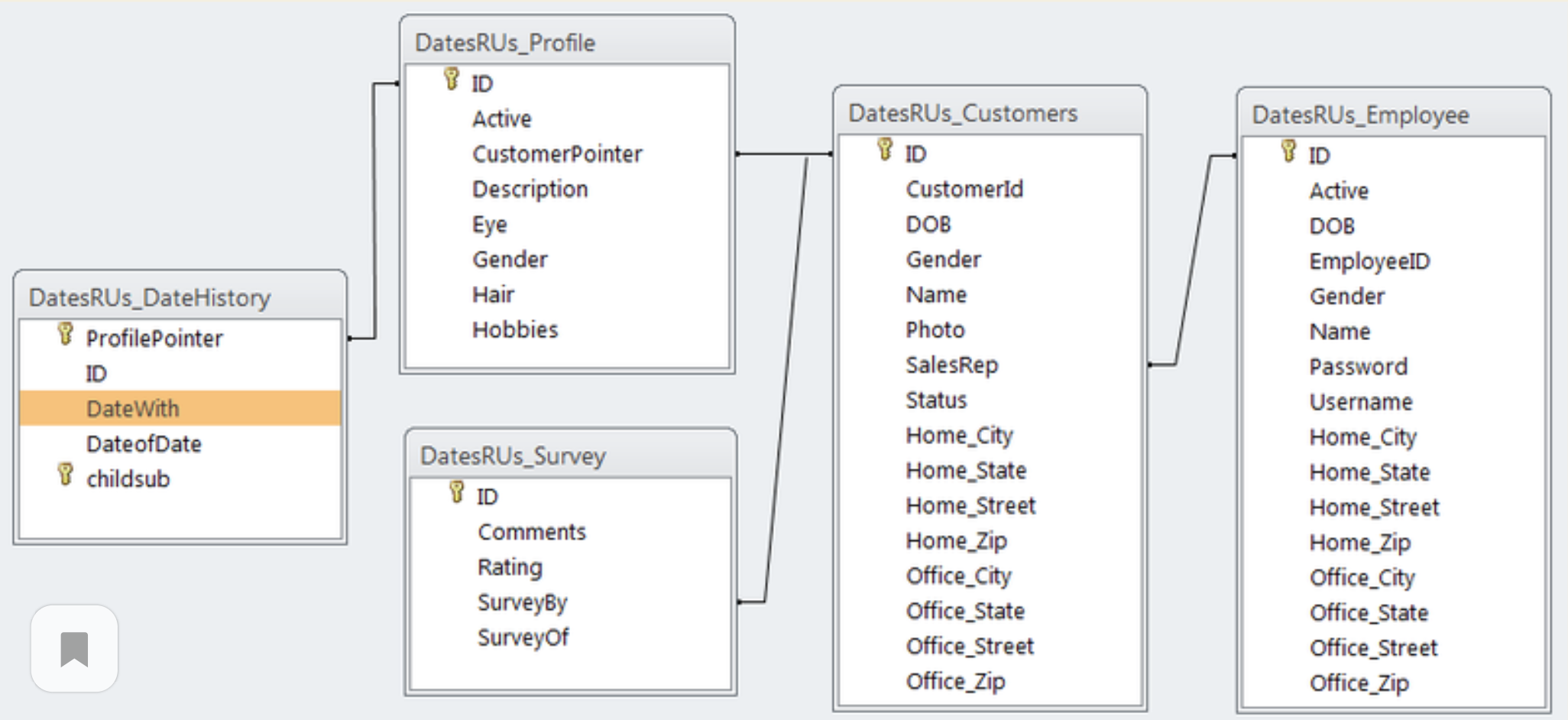
1. **Какой database engine используется в вашей СУБД?**

В основе Caché лежит ядро Caché Database Engine. Ядро базы данных сильно оптимизировано для наилучшей производительности, параллелизма, масштабируемости и надежности. Для достижения максимальной производительности, на каждой поддерживаемой платформе были проделаны колоссальные работы по оптимизации. Caché - полнофункциональная система баз данных; она включает в себя все функции, необходимые для запуска критически важных приложений (включая журналирование, резервное копирование и восстановление, а также средства администрирования системы). Чтобы сократить эксплуатационные расходы, Caché спроектирован таким образом, чтобы требовать значительно меньше администрирования баз данных, чем другие базы данных. Большинство развернутых систем Caché не имеют администраторов баз данных.

1. **Как устроен язык запросов в вашей СУБД? Разверните БД с данными и выполните ряд запросов.**

В Caché в качестве языка запросов используется SQL – диалект. SQL поддерживает полный набор возможностей реляционных СУБД, включая DDL, транзакции, ссылочную целостность, триггеры, хранимые процедуры, и поддерживает ODBC и JDBC (JDBC driver level 4). Команды и запросы SQL могут быть включены также в программы на языке Caché ObjectScript и в методы объектов Caché. Данные, доступные через SQL, представляются в виде таблиц со строками и столбцами. Так как Caché хранит данные в виде эффективных многомерных структур, достигается более высокая производительность SQL, чем в традиционных реляционных базах данных. Высокая производительность SQL проявляется как для скомпилированных в высокоэффективный код, так и при выполнении динамических (определяемых во время выполнения) SQL-выражений. В дополнение к стандартному синтаксису SQL, Caché поддерживает многие расширения языка, используемые в других СУБД, так что код многих приложений может быть перенесен в Caché без изменений, особенно те из них, что написаны с применением независимых от СУБД средств. В то же время, хранимые процедуры с применением синтаксиса, зависимого от производителей СУБД, требуют некоторой переработки с использованием средств, предлагаемых InterSystems. Caché SQL включает объектные расширения, упрощающие код SQL для записи и понимания.

Развернём бд, соотвествующую следующей схеме:



**Person:**  
Class DatesRUs.Person Extends (%Persistent, %Populate) {

/// Person’s name.  
Property Name As %String(POPSPEC = ".Name()");  
Index NameIndex On Name;

/// Person’s Date of Birth.  
Property DOB As %Date(POPSPEC = "Date()");

/// Person’s home address. This uses an embedded object.  
Property Home As Address;

/// Person’s office address. This uses an embedded object.  
Property Office As Address;

Property Gender As %String(DISPLAYLIST = ",Female,Male,", VALUELIST = ",F,M,");

Relationship Profiles As DatesRUs.Profile [ Cardinality = many, Inverse = CustomerPointer ];

}

**Address:**  
Class DatesRUs.Address Extends (%SerialObject, %Populate) {

/// The street address.  
Property Street As %String(MAXLEN = 80, POPSPEC = "Street()");

/// The city name.  
Property City As %String(MAXLEN = 80, POPSPEC = "City()");

/// The 2-letter state abbreviation.  
Property State As %String(MAXLEN = 2, POPSPEC = "USState()");

/// The 5-digit U.S. Zone Improvement Plan (ZIP) code.  
Property Zip As %String(MAXLEN = 5, POPSPEC = "USZip()");

}

**Profile:**  
Class DatesRUs.Profile Extends (%Persistent, %Populate) {

Property Hair As %String(DISPLAYLIST = ",Black,Blonde,Brown,Red,", VALUELIST = ",Bk,Be,Bn,R,");

Property Eye As %String(DISPLAYLIST = ",Brown,Blue,Green,Red,Yellow,", VALUELIST = ",Br,BL,G,R,Y,");

Property Gender As %String(DISPLAYLIST = ",Female,Male,", VALUELIST = ",F,M,");  
Index GenderIndex On Gender;

Property Hobbies As list Of %String(DISPLAYLIST = ",Jogging,Hiking,Reading,Skiing,Music,Video Games", VALUELIST = ",J,H,R,S,M,V,");

Property Description As %String(MAXLEN = 32000) [ InitialExpression = {##class(DatesRUs.Profile).Description()} ];

Relationship CustomerPointer As DatesRUs.Person [ Cardinality = one, Inverse = Profiles ];  
Index CustomerPointerIndex On CustomerPointer;

Relationship Datehistory As DatesRUs.DateHistory [ Cardinality = children, Inverse = ProfilePointer ];

Property Active As %Boolean;  
Index ActiveIndex On Active;

}

**Employee:**  
Class DatesRUs.Employee Extends (DatesRUs.Person, %Populate){

Property Username As %String;

Property Password As %String(COLLATION = "EXACT");

Property Active As %Boolean;

Property EmployeeID As %Integer [ InitialExpression = {$I(^DatesRUs.EmployeeC)} ];

Index EmpIDIndex On EmployeeID [ Unique ];

}

**Customers:**  
Class DatesRUs.Customers Extends (DatesRUs.Person, %Populate){

Property CustomerId As %Integer [ InitialExpression = {$I(^DatesRUs.CustomerC)} ];

Property Status As %String(DISPLAYLIST = ",Available, Dating, Married, Closed,", VALUELIST = ",A,D,M,C,");

Property SalesRep As DatesRUs.Employee;

Relationship Dates As DatesRUs.DateHistory [ Cardinality = many, Inverse = DateWith ];

Relationship SurveyByMe As DatesRUs.Survey [ Cardinality = many, Inverse = SurveyBy ];

Relationship SurveyOfMe As DatesRUs.Survey [ Cardinality = many, Inverse = SurveyOf ];

Property Photo As %Stream.GlobalBinary;

}

**DateHistory:**  
Class DatesRUs.DateHistory Extends (%Persistent, %Populate) {

Property DateofDate As %Date(MAXVAL = "03/18/2012", MINVAL = "01/01/2010");

Relationship DateWith As DatesRUs.Customers [ Cardinality = one, Inverse = Dates ];

Index DateWithIndex On DateWith;

Relationship ProfilePointer As DatesRUs.Profile [ Cardinality = parent, Inverse = Datehistory ];

}

**Survey:**  
Class DatesRUs.Survey Extends (%Persistent, %Populate) {

Property Rating As %Integer(MAXVAL = 10, MINVAL = 1);

Property Comments As %String(MAXLEN = 32000) [ InitialExpression = {##class(DatesRUs.Survey).Comment()} ];

Relationship SurveyBy As DatesRUs.Customers [ Cardinality = one, Inverse = SurveyByMe ];

Index SurveyByIndex On SurveyBy;

Relationship SurveyOf As DatesRUs.Customers [ Cardinality = one, Inverse = SurveyOfMe ];

//Index SurveyOfIndex On SurveyOf;

}

**Вставка 5000 строк в Customes:**

ClassMethod Method1() As %Integer {  
    for i=1:1:5000 {   
        set zip = "02134"     
        if (i#100=0) set zip = "02762"  
        set Rep = 113113  
        if ( i#2 = 0 ) set Rep = 113114  
        if ( i#3 = 0 ) set Rep = 113115  
          
        &sql(   
            INSERT INTO DatesRUs.Customers   
            (Name, Gender, Status, DOB, Home\_Zip, SalesRep)  
            VALUES (‘Test,Test’,‘M’,‘C’,+$H,:zip,:Rep)  
        )  
    }  
    Quit i  
}

**Выборка первых десяти записей из Customers по свойству унаследованному от родительского класса Person:**

set Statement1 = ##class(%SQL.Statement).%New()  
    set sql="SELECT TOP 10 Id, Name FROM DatesRUs.Customers WHERE Home\_Zip = ‘02762’ "  
    do Statement1.prepare( sql )  
    set Result1 = Statement1.%Execute()  
    while Result1.%Next() {  
       s id=Result1.%GetData(1), name=Result1.%GetData(2)  
    }

**Соединение таблиц Customers, Profile и Survey:**

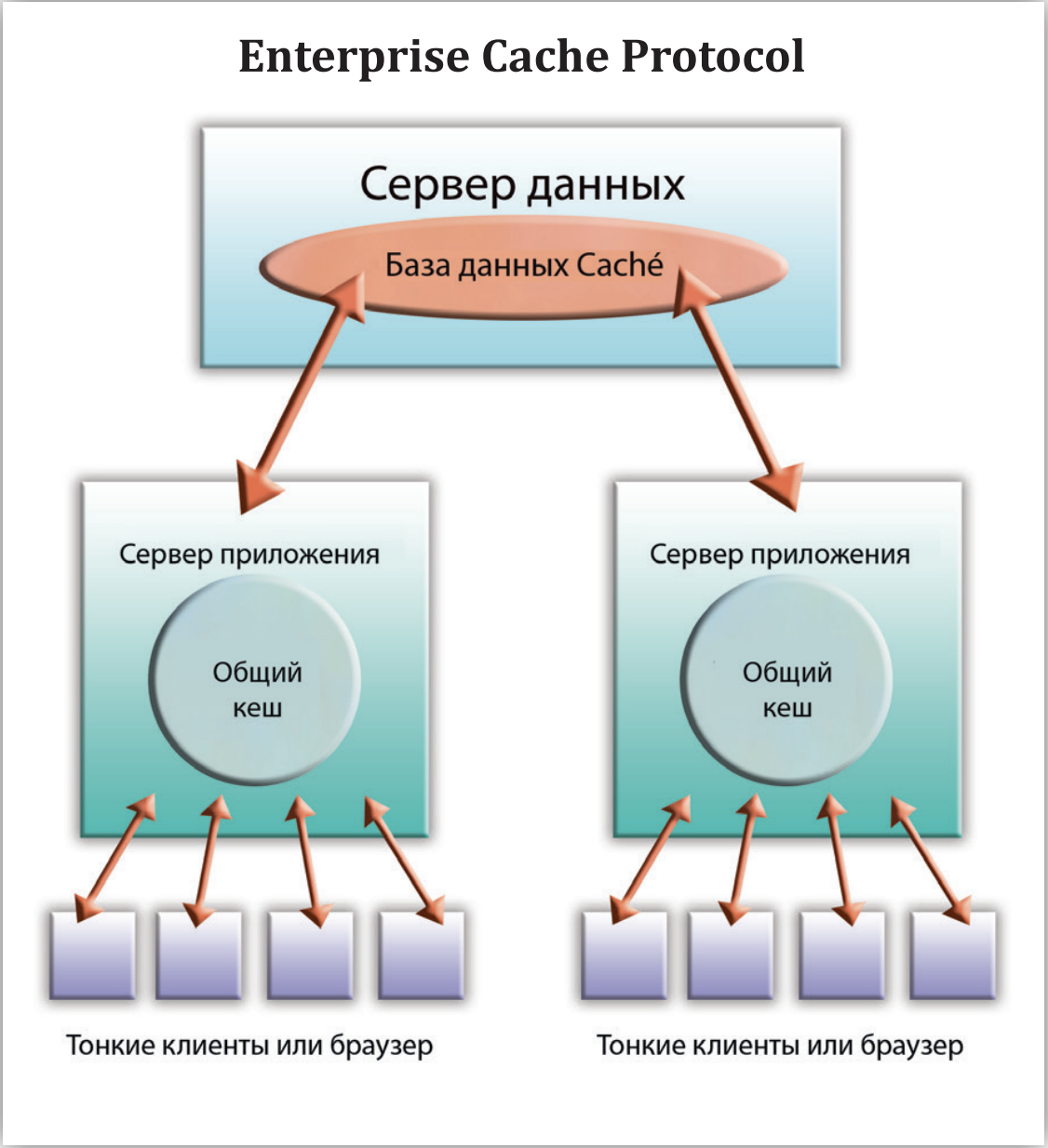
    set Statement1 = ##class(%SQL.Statement).%New()  
    set sql="SELECT C.Name,S.Rating "  
            \_" FROM DatesRUs.Customers C "  
            \_" JOIN DatesRUs.Profile P ON C.ID = P.CustomerPointer "  
            \_" LEFT OUTER JOIN DatesRUs.Survey S ON C.ID = S.SurveyOf "  
            \_" WHERE P.Hair=’R’ "  
    do Statement1.prepare(sql)  
    set Result1 = Statement1.%Execute()  
    while Result1.%Next() {  
        s id=Result1.%GetData(1), name=Result1.%GetData(2)  
    }

**Выборка из объединения таблиц Employee и Customers :**

        set Statement1 = ##class(%SQL.Statement).%New()  
        set sql="SELECT C.id, C.Name FROM DatesRUs.Employee E "  
                \_" JOIN DatesRUs.Customers C ON E.ID = C.SalesRep "  
                \_" WHERE EmployeeID = 113"  
        do Statement1.prepare(sql)  
        set Result1 = Statement1.%Execute()  
        while Result1.%Next() {  
                s id=Result1.%GetData(1), name=Result1.%GetData(2)  
        }

1. **Распределение файлов БД по разным носителям?**

В многокомпьютерных прикладных решениях СУБД Caché автоматически поддерживает распределенные кэш-блоки путем использования протокола Enterprise Cache Protocol (ECP). С использованием протокола ECP экземпляры Caché могут быть настроены как серверы данных и/или серверы приложений. Каждый элемент данных принадлежит серверу данных. Серверы приложений понимают, где находятся данные, и поддерживают локальные кэши недавно использовавшихся данных. Если сервер приложений не может удовлетворить запросы из локального кэша, он запрашивает требуемые данные на удаленном сервере данных. С помощью протокола ECP выполняется автоматическое управление согласованием кэш-памяти. Необходимые данные поддерживаются небольшим числом компьютеров, включенных в распределенную среду. Все, что требуется для заполнения такой среды данными, – это доступ по протоколу ECP к серверам данных. Они могут быть загружены в фоновом режиме, не затрагивая другие компьютеры, используемые в это время для выполнения других задач. Когда серверы включаются в рабочий процесс, их кэш-памяти автоматически перезаполняются данными из запрошенных.

****

1. **На каком языке/ах программирования написана СУБД?**

InterSystems Caché основан на языке программирования MUMPS.

1. **Какие типы индексов поддерживаются в БД? Приведите пример создания индексов.**

В Caché можно использовать оба вида индексов ― традиционные и транзакционные битовые индексы. Кроме того, здесь поддерживаются многоколоночные индексы.

Пример создания индексов: CREATE INDEX StateIdx ON TABLE Sample.Person (Home\_State)

Определение индексов при создании классов выглядит например так:

Class MyApp.Employee Extends %Persistent [DdlAllowed]

{

Property Name As %String;

Property Salary As %Integer;

Property State As %String(MAXLEN=2);

Index MainIDX On(State,Salary);

}

1. **Как строится процесс выполнения запросов в вашей СУБД?**

Caché SQL автоматически использует оптимизатор запросов для создания плана запросов, который обеспечивает оптимальную производительность запросов в большинстве случаев. Этот оптимизатор улучшает производительность запросов различными способами, включая определение индексов использования, определение порядка оценки нескольких условий AND, определение последовательности таблиц при выполнении нескольких соединений и многие другие операции оптимизации. Caché поддерживает следующие инструменты для оптимизации SQL-запросов:

* [SQL Runtime Statistics](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_optquery#GSQLOPT_optquery_stats)для генерации статистики производительности при выполнении запроса.
* [Анализатор индексов](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_optquery#GSQLOPT_optquery_ind)для отображения различных отчетов анализатора индексов для всех запросов в текущем пространстве имен. Это показывает, как Caché SQL собирается выполнять запрос, давая вам общее представление о том, как используются индексы. Этот анализ индекса может указывать на то, что для повышения производительности следует добавить один или несколько индексов.
* [Показать план](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_optquery#GSQLOPT_optquery_showplan), чтобы отобразить оптимальный план выполнения (по умолчанию) для SQL-запроса.
* [Альтернативное отображение планов](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_optquery#GSQLOPT_optquery_altshowplans)для отображения доступных альтернативных планов выполнения SQL-запроса со статистикой.
* [Опции оптимизации индекса](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_optquery#GSQLOPT_optquery_from)доступны ИЗ пунктов, регулирующих все условия, или %NOINDEX, предваряющий индивидуальное условие.
* Доступна [параллельная обработка запросов](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_optquery#GSQLOPT_optquery_parallel)ключевое слово %PARALLEL FROM позволяет многопроцессорным системам разделить выполнение запросов между процессорами
* [Кэшированные запросы](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_cachedqueries), чтобы включить повторный запуск динамических SQL-запросов без накладных расходов на подготовку запроса каждый раз при его выполнении.
* [SQL-операторы](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_sqlstmts)для сохранения наиболее скомпилированного встроенного SQL-запроса. В главе «Инструкции SQL и замороженные планы».
* [Замороженные планы](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_frozenplans)для сохранения определенной компиляции встроенного SQL-запроса. Эта компиляция используется, а не более поздняя компиляция. В главе «Инструкции SQL и замороженные планы».

Следующие инструменты используются для оптимизации табличных данных и, таким образом, могут оказать значительное влияние на все запросы, выполняемые с этой таблицей:

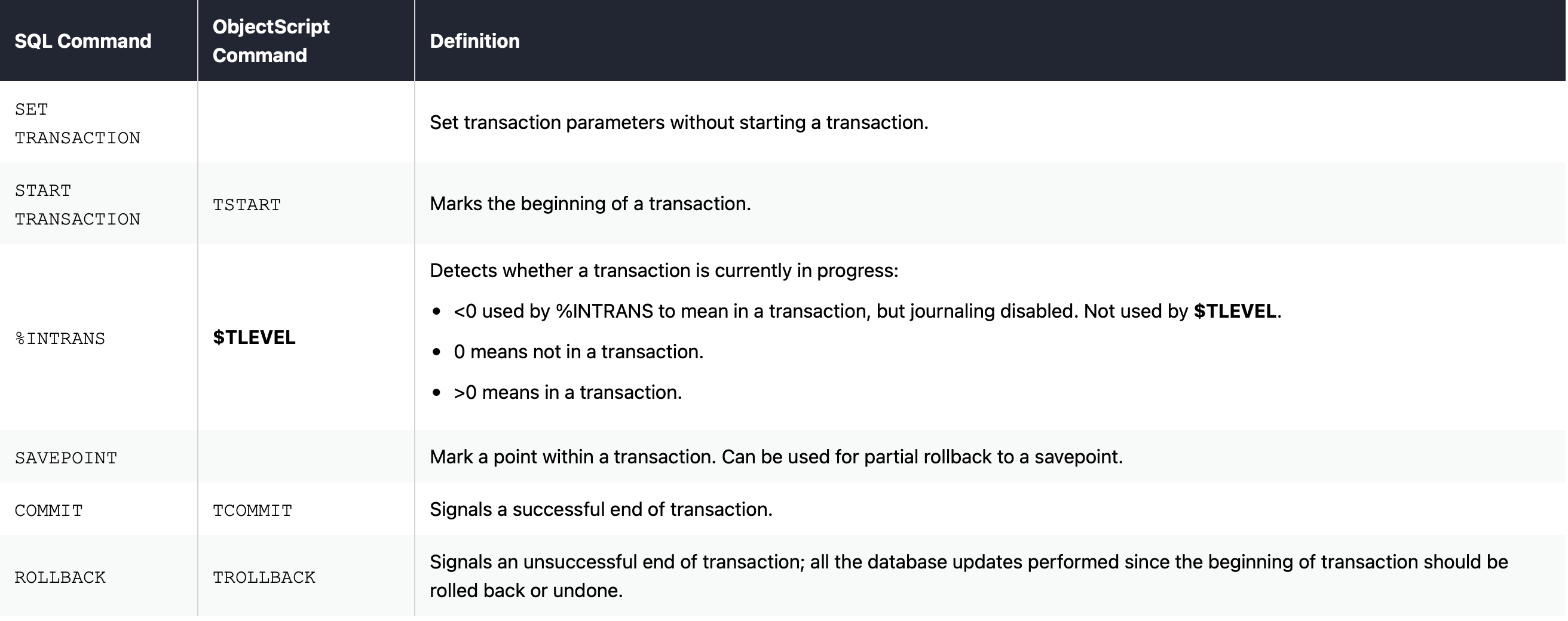
* [Определение индексов](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_indices)может значительно ускорить доступ к данным в определенных индексированных полях.
* [ExtentSize, Selectivity и BlockCount](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_opttable#GSQLOPT_opttable_query_opt)для указания оценок табличных данных перед заполнением таблицы данными; эти метаданные используются для оптимизации будущих запросов.
* [Настройте таблицу](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_opttable#GSQLOPT_opttable_tunetable)для анализа репрезентативных табличных данных в заполненной таблице; эти сгенерированные метаданные используются для оптимизации будущих запросов.

1. **Есть ли для вашей СУБД понятие «план запросов»? Если да, объясните, как работает данный этап.**

Компиляция SQL-запроса дает набор инструкций для доступа и возврата данных, указанных в запросе. Эти инструкции выражены в виде кода ObjectScript. На инструкции и последовательность их выполнения влияют данные компилятора SQL о структуре и содержании таблиц, участвующих в запросе. Компилятор пытается использовать такую информацию, как размеры таблиц и доступные индексы, чтобы сделать набор инструкций максимально эффективным. План доступа к запросам ([ShowPlan](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSQLOPT_optquery" \l "GSQLOPT_optquery_showplan)) - это читаемый человеком перевод этого результирующего набора инструкций. Автор запроса может использовать этот план доступа к запросу, чтобы увидеть, как будут доступны данные. В то время как компилятор SQL пытается максимально эффективно использовать данные, указанные в запросе, иногда автор запроса знает больше о каком-то аспекте хранимых данных, чем очевидно компилятору. В этом случае автор может использовать план запроса для изменения исходного запроса, чтобы предоставить компилятору запроса дополнительную информацию или больше указаний. Результатом «ShowPlan» является серия заявлений о том, какая обработка будет выполнена для доступа и представления данных, указанных в исходном запросе.

1. **Поддерживаются ли транзакции в вашей СУБД? Если да, то расскажите о нем. Если нет, то существует ли альтернатива?**

Да, InterSystem Cache поддерживает транзакции. В Caché вы определяете транзакции в приложениях, используя либо SQL-операторы в процедурах источника макросов, либо команды ObjectScript. Оба метода работают, независимо от того, выполняются ли изменения базы данных. Оба метода работают, независимо от того, выполняются ли изменения базы данных, составляющие транзакции, с помощью инструкций SQL INSERT, UPDATE и DELETE или команд ObjectScript SET и KILL. Caché поддерживает операции ANSI SQL COMMIT WORK и ROLLBACK WORK (в Caché SQL ключевое слово WORK является необязательным). Он также поддерживает расширения Caché SQL SET TRANSACTION, START TRANSACTION, SAVEPOINT и %INTRANS. Кроме того, Caché реализует некоторые команды транзакций, которые являются частью стандарта M Type A.



Эти команды ObjectScript и SQL полностью совместимы и взаимозаменяемы, за следующим исключением:

ObjectScript TSTART и SQL START TRANSACTION запускают транзакцию, если ни одна транзакция не является текущей. Однако START TRANSACTION не поддерживает вложенные транзакции. Поэтому, если вам нужны (или могут понадобиться) вложенные транзакции, предпочтительнее начать транзакцию с TSTART. Если вам нужна совместимость со стандартом SQL, используйте START TRANSACTION.

**Использование LOCK в транзакциях.**

Всякий раз, когда вы получаете доступ к глобальному, к которому может получить доступ более одного процесса, вам необходимо защитить целостность базы данных с помощью команды [LOCK](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOS_clock)на этом глобальном. Вы выдаете блокировку, соответствующую глобальной переменной, изменяете значение глобальной, а затем разблокируете блокировку. Команда LOCK используется как для блокировки, так и для разблокировки указанной блокировки. Другие процессы, желающие изменить значение глобального запроса, блокировка, которая ждет, пока первый процесс не снимет блокировку.

Существует три важных соображения при использовании замков в транзакциях:

* Операции блокировки/разблокировки не откатываются.
* В рамках транзакции, когда вы разблокируете блокировку, удерживаемую процессом, может произойти одна из двух вещей: 1) Замок немедленно разблокирован. Блокировка может быть немедленно получена другим процессом; 2) Блокировка помещена в состояние блокировки. Блокировка разблокирована, но не может быть получена другим процессом до конца текущей транзакции. Если блокировка находится в состоянии разблокировки, Caché откладывает разблокировку до тех пор, пока транзакция не будет зафиксирована или откат. В рамках транзакции блокировка, похоже, разблокирована, что позволяет последующую блокировку того же значения. Однако вне транзакции блокировка остается заблокированной.
* Операции блокировки, по тайм-ауту, установленные [$TEST](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOS_vtest). Значение, установленное в **$TEST**во время транзакции, не откатывается.

**Откат транзакции в приложении.**

Если вы столкнулись с ошибкой во время транзакции, вы можете откатить ее тремя способами:

* Выполните команду SQL rollback, ROLLBACK WORK
* Выполните команду отката ObjectScript, TROLLBACK
* Выполните вызов **%ETN**

Разработчики приложений могут использовать два типа команд отката для обозначения неудачного конца транзакции и автоматического отката неполных транзакций:

* Используйте ##sql(ROLLBACK WORK) в исходной процедуре макросов.
* Используйте команду ObjectScript TROLLBACK в макросе или промежуточном исходном коде.

Откат транзакции происходит автоматически во время:

* Запуск Caché, если требуется восстановление. Когда вы запускаете Caché и определяете, что восстановление необходимо, любая неполная транзакция на компьютере будет откатиться.
* Окончание процесса с помощью команды [HALT](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOS_chalt)(для текущего процесса) или утилиты [^RESJOB](https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOS_chalt#RCOS_chalt_resjob)(для других процессов). Остановка фонового задания (неинтерактивный процесс) автоматически откатывает изменения, внесенные в текущую незавершенную транзакцию. Остановка интерактивного процесса запрашивает, следует ли фиксировать или откатывать изменения, внесенные в текущую незавершенную транзакцию. Если вы выдаете ^RESJOB в пользовательском процессе в режиме программиста, система выведет пользователю сообщение с вопросом, хочет ли он, чтобы текущая транзакция была зафиксирована или откат.

Кроме того, системные менеджеры могут откатывать неполные транзакции в базах данных кластера, запустив утилиту **^JOURNAL**. При выборе опции «**Восстановить глобальные данные из журнала**» в главном меню утилиты **^JOURNAL**файл журнала восстанавливается, а все неполные транзакции откатываются.

Каждый экземпляр Caché ведет журнал. Журнал представляет собой набор файлов, в котором хранится журнал изменений, внесенных в базу данных с момента последнего резервного копирования. Обработка транзакций Caché работает с ведением журнала для поддержания логической целостности данных. Журнал содержит операции SET и KILL для глобальных переменных в транзакциях независимо от настройки журнала баз данных, в которых находятся затронутые глобальные переменные, а также все операции SET и KILL для глобальных переменных в базах данных, **состояние глобального журнала**которых вы установили в «Да». Резервное копирование может быть выполнено во время обработки транзакций; однако результирующий файл резервной копии может содержать частичные транзакции. В случае аварии, требующей восстановления из резервной копии, сначала восстановите файл резервной копии, а затем примените файлы журнала к восстановленной копии базы данных. Применение файлов журнала восстанавливает все журналируемые обновления с момента резервного копирования до момента аварии. Применение журналов необходимо для восстановления целостности транзакций вашей базы данных путем завершения частичных транзакций и отката незафиксированных транзакций, так как базы данных, возможно, содержали частичные транзакции на момент резервного копирования.

1. **Какие методы восстановления поддерживаются в вашей СУБД. Расскажите о них.**

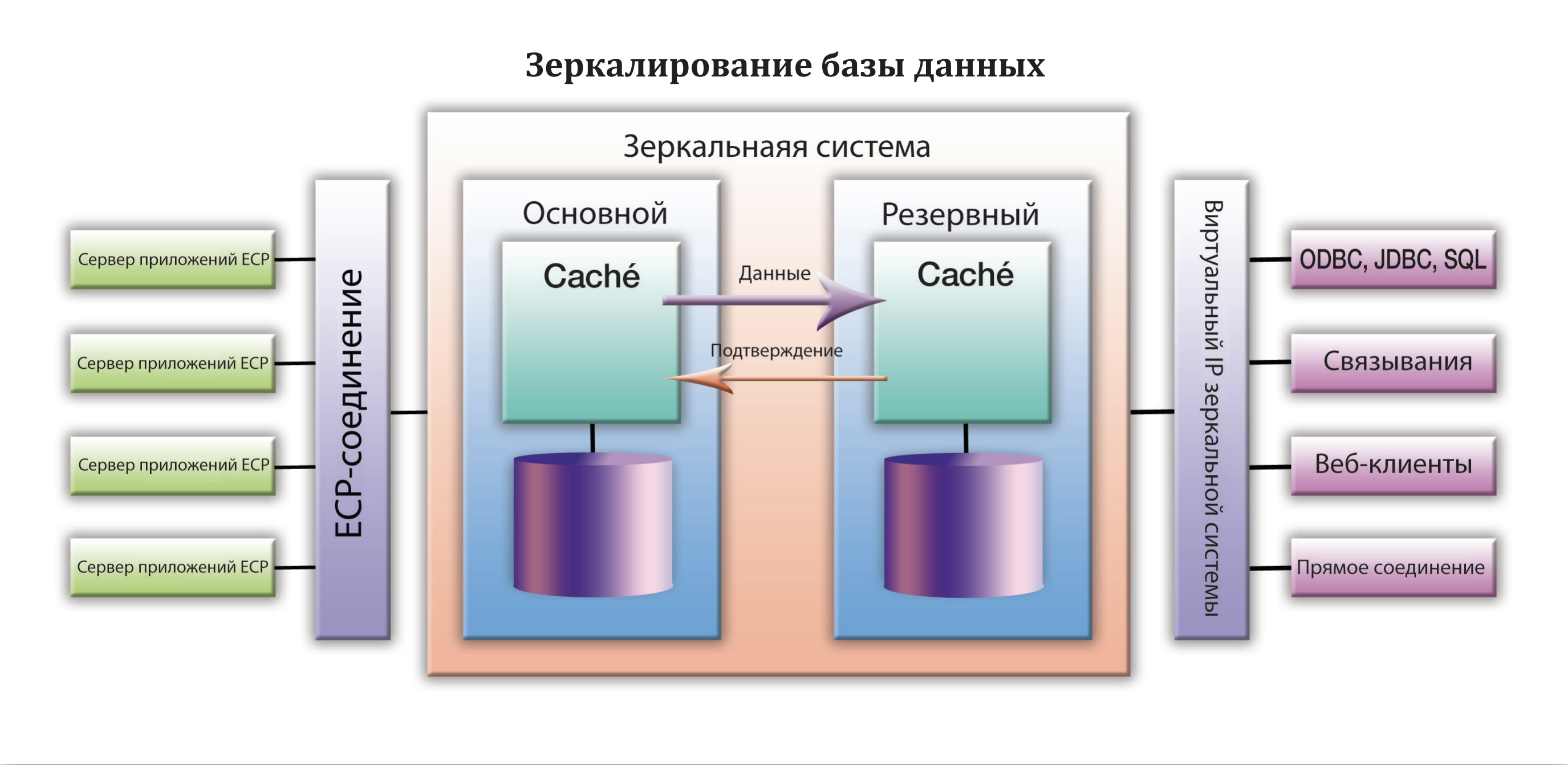
Функция журналирования (Caché Write-Image Journaling) и другие средства гарантируют сохранение целостности базы данных в большинстве случаев (включая отказ электропитания) и быстрое восстановление после сбоя с минимальными последствиями для пользователей.

Caché предоставляет различные возможности по обеспечению отказоустойчивости, высокой готовности и по сохранению состояния баз данных при сбоях. Существуют варианты конфигураций, уменьшающие влияние сбоев на пользователей системы:

* технология зеркалирования
* распределенный протокол ECP
* Failover кластеры

**Зеркалирование базы данных.**

Для поддержания системы в актуальном состоянии зеркалированные базы данных основного и резервного узлов синхронизируются в реальном масштабе времени. Синхронизация осуществляется через сеть таким образом, чтобы минимизировать влияние данного процесса на производительность основного узла. Зеркалированные базы данных изменяются только на основном узле; все зеркалированные базы данных в системе, определенной в качестве резервного узла, поддерживаются в режиме «только для чтения», что предотвращает случайные обновления этих БД



Внешние клиенты (обращения через язык программирования, ODBC/JDBC/SQL-клиенты, пользователи с прямым подключением и т.д.) подключаются к системе зеркалирования по ее виртуальному IP-адресу − Mirror Virtual IP (VIP), который задается в процессе настройки системы. Этот адрес автоматически присваивается интерфейсу узла, назначенного системой зеркалирования в качестве основного в данный момент. Указание VIP-адреса необязательно: если он не задан, все внешние клиенты должны соединяться непосредственно с работающим основным узлом и «знать» оба узла и роли, заданные им на данный момент в составе системы зеркалирования.

**Использование зеркального отображения БД с протоколом ECP.**

Серверы приложений, поддерживающие протокол InterSystems Enterprise Cache Protocol (ECP), автоматически получают информацию об обоих узлах системы зеркалирования, в том числе о том, какой из них в данный момент является основным. Если основной узел выйдет из строя, серверы приложений ECP воспримут переход на другой узел просто как перезапуск ECP сервера данных ― серверы приложений просто установят соединение заново и продолжат работу. В крайнем случае все, с чем могут столкнуться пользователи при сбое, – это небольшая пауза в работе.

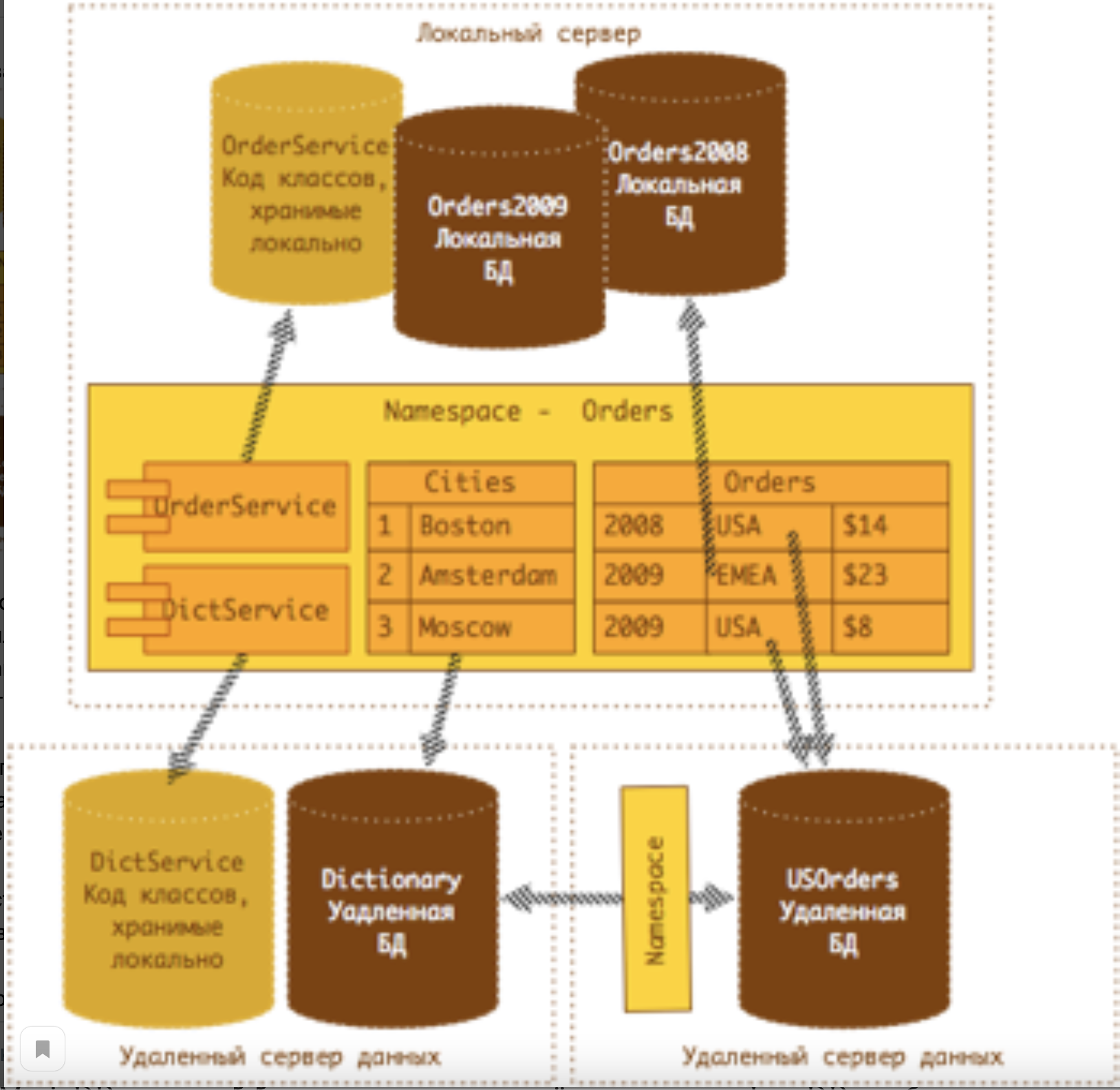
**Failover кластеры.**

Один из распространенных вариантов кластерных конфигураций ― failover. Такая конфигурация обеспечивает доступ нескольких серверов к одной базе данных, при этом только на одном из них выполняется Caché. Если происходит отказ активного сервера, Caché стартует на другом сервере, который становится активным. Пользователи сразу же могут подключаться к новому серверу. Сервер данных ECP может быть сконфигурирован как отказоустойчивый кластер. Если основной сервер данных выйдет из строя, вместо него, «заняв его место», будет функционировать резервный сервер данных.

1. **Расскажите про шардинг в вашей конкретной СУБД. Какие типы используются? Принцип работы.**

C одной стороны Caché не имеет готовых out of the box вариантов с такими названиями как как шардинг и партиционирование. С другой точки зрения это не совсем так, потому что в Caché либо это делается немного иначе, либо опять же, так же как и в случае с глобалами, предоставляется надежные базовые технологии для эффективного обеспечения таких возможностей. С использованием ECP, концепции областей (namespace), Subscript Level Mapping возможна реализация эффективной распределенной обработки данных.

Партиционирование и распределенное хранение с помощью ECP и SLM:



1. **Возможно ли применить термины Data Mining, Data Warehousing и OLAP в вашей СУБД?**

InterSystem Cache может использоваться для аналитики больших данных. СУБД поддерживает соединение с инструментами, применяемыми в соответствующем назначении, например с Apache Spark. Платформа **Caché** поддерживает использование большого количества моделей, благодаря чему она идеально подходит для моделирования реальных данных со сложной структурой. Уникальная возможность анализа “снизу вверх” позволяет выполнять анализ неструктурированных текстовых данных без использования заранее созданных библиотек.

1. **Какие методы защиты поддерживаются вашей СУБД? Шифрование трафика, модели авторизации и т.п.**

Безопасность в системе Caché обеспечивается за счет выполнения аутентификации, авторизации, аудита и шифрования БД. Предоставляя пользователям эти возможности, Caché позволяет минимизировать затраты на поддержание высокой производительности приложений в системе.

Аутентификация в Caché ― процедура проверки подлинности пользователя ― выполняется как для реальных пользователей, так и для систем и даже устройств. В Caché может применяться ряд механизмов аутентификации, которые перечислены ниже:

* Протокол сетевой аутентификации Kerberos: Является наиболее надежным средством аутентификации. Kerberos Authentication System обеспечивает математически проверенную, надежную сетевую аутентификацию пользователей.
* Протокол LDAP: Caché может обеспечить проведение аутентификации по протоколу Ligthweight Directory Access Protocol (LDAP). В этом случае Caché запрашиваетLDAP-сервер для аутентификации пользователя, основываясь на информации своей системной базы данных пользователя. Через LDAP-сервер также могут контролироватся все аспекты управления паролями, политики паролей и т.п.
* Пароли: Caché подсказывает пользователю о необходимости указания пароля и сравнивает хэш пароля с хэш-значением, которое было в ней сохранено.
* Делегированная аутентификация: Делегированная аутентификация (Delegated Authentication) применяется как средство создания индивидуально настраиваемых аутентификационных механизмов. Разработчик приложения полностью контролирует логику кода делегированной аутентификации. Caché включает в себя шаблон для построения кода аутентификации уполномоченным пользователем.
* Аутентификация на базе операционной системы: В этом случае аутентификация основана на уверенности в том, что операционная система проверила идентичность каждого пользователя и что в ней применяются те же аутентификационные данные, которые требуются для работы с Caché

В Caché имеются встроенные средства поддержки проведения двухфакторной аутентификации, которая предусматривает необходимость подтверждения пользователями их идентичности на основе чего-то, что они знают, и чего-то, что они имеют.

После выполнения аутентификации пользователя следующей операцией по обеспечению безопасности является определение того, что этому пользователю разрешено применять, просматривать и редактировать. Определение прав пользователя и контроль пользовательского доступа называется авторизацией. Назначение и контроль прав пользователей обычно осуществляются через Caché Management Portal.

Многим приложениям необходим надежный аудит, особенно это касается тех, к которым предъявляются требования распоряжений правительства или законов, например HIPAA или закон Сарбанеса-Оксли (Sarbanes-Oxley). Все события системы Caché и приложений записываются в журнал, доступный только для добавления и совместимый с любым инструментом для генерации отчетов по SQL-запросам.

Шифрование данных в местах хранения (Data-at-Rest) Caché поддерживает два типа шифрования баз данных:

* Администратор безопасности может определить один или более файлов CACHE.DAT, которые требуется шифровать. После этого все содержимое этих файлов шифруется, включая все индексы.
* Разработчики приложений могут использовать системные функции для шифрования / дешифрования данных, которые затем могут быть переданы или сохранены в БД. Этот способ может применяться для шифрования важных данных, чтобы защитить их от пользователей, имеющих доступ лишь к содержимому базы данных только для чтения, но не имеющих ключа.

По умолчанию в Caché данные шифруются с применением AES (Advanced Encryption Standard) – симметричного алгоритма, использующего ключи длиной 128, 192 или 256 бит. Ключи шифрования хранятся в защищенной области памяти. Caché предоставляет все необходимые возможности для управления ключами шифрования. Журналы также могут шифроваться.

Шифрование для обеспечения безопасности каналов передачи данных (Datain-Motion) Caché поддерживает протокол безопасности SSL (Secure Sockets Layer) и являющийся его преемником протокол TLS (Transport Layer Security) для обеспечения безопасных соединений нескольких типов, описанных ниже :

* Когда система Caché функционирует в качестве сервера, она принимает данные о соединениях и организует использование SSL, включая соединения теневых приемников Caché (shadow destinations) и теневых источников Caché (shadow sources).
* Когда Caché работает в качестве клиента, она может соединяться с серверами, требующими применения SSL.

Во всех случаях Caché использует конфигурацию SSL/TLS Configuration, которая определяет различные характеристики для работы сервера Caché с SSL/TLS-соединениями.

1. **Какие сообщества развивают данную СУБД? Кто в проекте имеет права на коммит и создание дистрибутива версий? Расскажите об этих людях и/или компаниях.**

InterSystem Developer community развивают данную СУБД. Корпорация InterSystems была основана в 1978 году.  Компания создает платформы для управления данными, инструменты стратегической совместимости и средства аналитики, применяемые в здравоохранении, в сфере финансовых услуг, в аппарате государственного управления и во многих других областях. Продукты InterSystem включают в себя систему управления базой данных Caché, платформу интеграции Ensemble, информационную платформу HealthShare для медицинской информации и информационную систему TrakCare по вопросам здравоохранения, которая продается за пределами США. InterSystems базируется в Кембридже, штат Массачусетс.

<https://community.intersystems.com/tags/caché>

1. **Создайте свои собственные данные для демонстрации работы СУБД.**

Создадим класс :

/// Место проведения представления

Class place.One Extends %Persistent

{

///Название места проведения представления

Property Name As %String;

///Описание

Property Description As %String;

///Детали

Property Details As %String;

}

Теперь создадим экземпляр класса и заполним его:

Узел: papahost, Экземпляр: TRYCACHE

USER>zn “BS”

BS>set obj= **##class(place.One).%New()** // Не заботимся о конструкторе

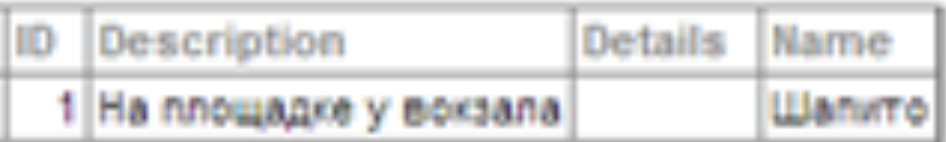
BS>**set obj.Name = "Шапито"** // Не заботимся о сеттерах-геттерах

BS>**set obj.Description = "На площадке у вокзала"**

BS>**do obj.%Save()**// Не заботимся о том, как происходит сохранение данных

Используем SQL доступ: select \* from place.One

Результат:



1. **Как продолжить самостоятельное изучение языка запросов с помощью демобазы? Если демобазы нет, то создайте ее.**

<https://www.intersystems.com/try-intersystems-iris-for-free/>

<https://github.com/intersystems-community/irisdemo-demo-htap>

1. **Где найти документацию и пройти обучение?**

https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=SETAppDev

<http://student-telecom.ru/_ld/0/24_144_NSUTextBook.pdf>

http://writeimagejournal.com/?page\_id=1779

1. **Как быть в курсе происходящего?**

https://www.intersystems.com/news-events/news/?filter[news-type]=press-releases