Инженерный вестник Дона, №11 (2022)
ivdon.ru/ru/magazine/archive/n11y2022/8009

Реализация временных моделей для темпоральных надстроек системы управления баз данных

С.В. Сафонов

Московский технический университет связи и информатики, г. Москва
Агентство ВПС-Мониторинг, г. Москва

Аннотация: В настоящее время ведутся попытки реализации темпоральной системы управления базами данных, учитывающие все возможности темпоральной поддержки. В представленной статье рассматриваются способы построения темпоральных баз данных. На примере создания темпоральной базы данных для организации, занимающейся проектной деятельностью, показана реализация временной модели для темпоральных надстроек системы управления баз данных. Разработана диаграмма сущностей, отражающая таблицы базы данных и связи между ними. Созданы классы сущностей, соответствующие таблицам из баз данных в отдельных файлах. Представлены методы, транслирующие темпоральные запросы в реляционные.
Ключевые слова: система управления базами данных, темпоральная база данных, темпоральные запросы, реляционные модели баз данных, темпоральные модели баз данных, темпоральная надстройка.

Темпоральные базы данных представляют собой базы данных с поддержкой временного фактора. Отличительной особенностью темпоральных баз данных является хранение и обработка данных, связанных с определёнными датами или промежутками времени, с учётом известного правила интерпретации временных интервалов для конкретной системы управления базами данных (СУБД) [1]. Темпоральные базы данных нашли своё применение в различных сферах человеческой деятельности, например в целях построения модели знаний на предприятии [2] или в целях диагностирования устройств железнодорожной автоматики [3] и др.

Существует несколько подходов для представления времени – с помощью временных точек и временных интервалов. Представление времени с использованием временных интервалов на основании темпоральной логики представлено в работе [4] (построение множества отношений для отражения всех возможных и взаимно различных отношений, которые могут иметь место между двумя заданными темпоральными интервалами). Точечный

подход представлен в работе [5] (представление времени с помощью временных точек и трех основных операций “точечной” алгебры: <,>, =). В работе [6] рассматривается временная логика предикатов, предназначенная для изучения специфических областей. Описана сигнатура временной теории, содержащая, кроме глобальных констант, локальные индивидные константы и локальные высказывания. Из всех перечисленных подходов к учету фактора времени в настоящей работе будем использовать интервальный подход к представлению времени и реализации его в темпоральной модели данных.

Для поддержки темпоральных данных, используемых в СУБД,, известно несколько подходов [7]:

преобразование ядра СУБД с изменением его структуры, в результате чего реализуется абсолютная поддержка темпоральных данных;

создание библиотеки, преобразующей запросы в SQL-запросы на языках SQL/Temporary и др.

построение темпорального приложения при работе с реляционной СУБД;

расширение реляционной модели данных временными атрибутами путем надстройки над реляционной СУБД.

Целью настоящей работы является реализация темпоральной модели данных в базе данных (БД) компании, осуществляющей проектную деятельность. Вначале необходимо определить цели моделирования и выделить основные сущности, находящиеся в пределах моделируемой проблемной области.

Целью моделирования является разработка схемы базы данных для организации, занимающейся разработкой проектов. Основными информационными единицами являются Подрядчик (Contractor), Работник (Workers), Проекты компании (Laout), Должность (Post) и Доход (Income).

Каждая выделенная сущность характеризуется своим именем и определением (назначением).

В дальнейшем определяются связи между сущностями и атрибутами сущностей. Значение первичного ключа должно являться уникальным, поэтому атрибуты, на которые ссылается внешний ключ, тоже будут уникальными.

Первая сущность – Contractor. Данная таблица хранит информацию обо всех организациях, с которыми сотрудничает рассматриваемая проектная компания. В качестве атрибутов выступают:

Contractor _id – первичный ключ данной таблицы,

Contractor _name – название организации,

Contractor _establish – дата регистрации организации,

Contractor _adress – адрес организации,

Contractor _ Contractor _director – управляющий организации (внешний ключ - FK).

Таблица Income – в ней хранится подробная информация о доходах организации. Таблица содержит следующие атрибуты:

Income_id - первичный ключ таблицы,

Income_month – месяц,

Income_year – год,

Income_income – доход за месяц,

Income_spend – расход за месяц,

Income_profit –прибыль за месяц,

Income_transact_time – дата занесения информации в БД,

Company_Company_id – id компании.

Таблица Workers – данная таблица хранит в себе данные о работниках организации. Набор атрибутов:

Workers_id – первичный ключ таблицы,

Workers_name – имя работника,

Workers_surname – фамилия работника,

Workers_patronymic – отчество работника,

Workers_civility – пол работника,

Workers_adress – адрес работника,

Compamy_company_id – id компании.

Таблица Laout – темпоральная таблица, содержит информацию о проектах компаний. Атрибуты:

Laout_id – PK таблицы,

Laout_name – название,

Laout_start – дата начала,

Laout_end –планируемая дата завершения,

Laout_cost – стоимость,

Laout_transact_time – дата внесения информации в БД.

Таблица Position_list – хранит информацию обо всех должностях в компании, в том числе за предыдущий период. Атрибуты таблицы:

Position_list_id - первичный ключ таблицы,

Position_list_name – название должности.

Таблица Post – таблица с темпоральной поддержкой, содержащая информацию о занимаемых должностях каждого бывшего и настоящего работника в организации – дата начала и окончания работы сотрудника. Атрибуты:

Post_position_id – первичный ключ таблицы,

Post_position_start – дата начала работы в данной должности,

Post_position_end – дата, с которой работник не работает в данной должности,

Post_position_transact_time – дата внесения информации в БД,

Employee_Employee_id – id сотрудника (занимающего должность,

Position_Position_id – id должности.

Таблица Income_temporal – темпоральная таблица, предоставляющая данных с временной поддержкой обо всех окладах и их изменениях за различные периоды. В таблице содержатся следующие атрибуты:

Income_temporal _id – первичный ключ таблицы,

Income_temporal _amount – оклад работника,

Income_temporal _start – дата, с которой работнику назначен указанный оклад,

Income_temporal _end – дата, с которой оклад работника не изменен,

Income_temporal _transact_time – дата занесения информации в БД,

Workers_ Workers _id – id сотрудника.

В целях определения связей между сущностями при помощи программы MySQL Workbench построена диаграмма сущностей и отношения между ними, которые схематично изображены на рисунке 1.

Связь между таблицами Contractor и Income означает, что информация, содержащаяся в данных таблицах, хранится за каждый месяц. Связь между таблицами Workers и Income temporal подразумевает всю историю окладов работников, находящихся в штате компании. Связь, соединяющая таблицы Workers и Post, подразумевает, что у каждого работника в таблице Post хранится вся история его должностей в данной организации. Связь между таблицами Position list и Post означает, что одну и ту же должность в организации в различное время замещают различные люди.

Для организации доступа к БД необходимо организовать использование программной платформы, подключаемая к нашей БД. В файле, организующем доступ к данным необходимо указать соединения, служебную информацию и ресурсы, которые будут использоваться из БД. После организации доступа создаются классы сущностей, которые будут

соответствовать таблицам из БД. Кроме того, все связи указываются в отдельных xml файлах.

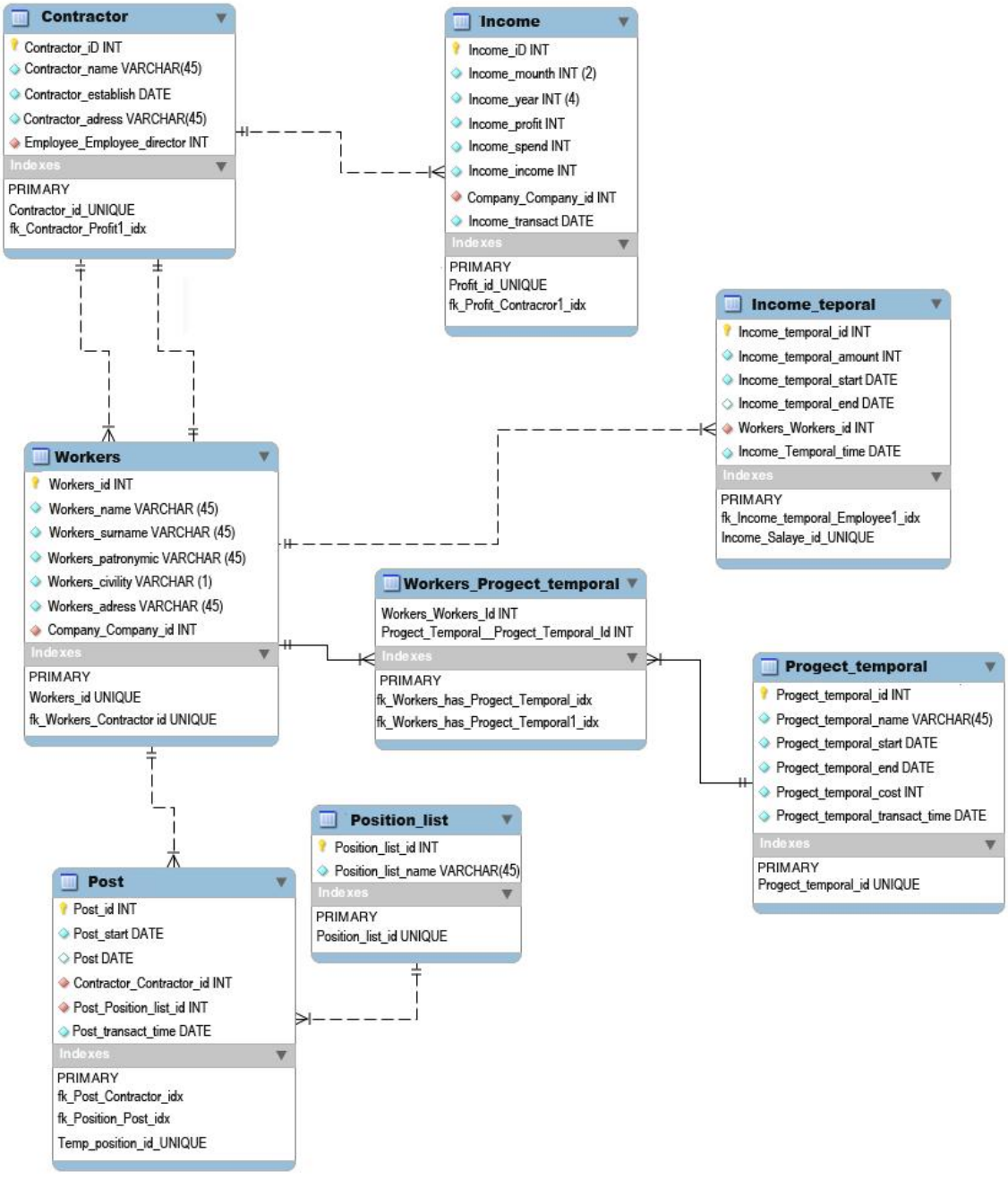

Рисунок 1. – Диаграмма сущностей и их отношений БД проектной компании

Затем реализуется темпоральная алгебра [8], целью которяй является реализация обработчика, преобразующего входные данные в SQL-запрос.

Интерфейс темпоральной алгебры будет состоять из 7-ми методов, изображенных на рисунке 2.

```
1    public interface TempAlgebraDao {
2       List<Object> selection(Object table, String condition);
3       List<Object> projection(Object table, Date valid_begin, Date valid_end);
4       List<Object> union(Object r1, Object r2);
5       List<Object> intersection(Object r1,Object r2);
6       List<Object> difference(Object r1,Object r2);
7       List<Object> composition(Object r1,Object r2);
8       List<Object> aggregation(Object r, String agg);
9    },
```

Рисунок 2. – Код интерфейса темпоральной алгебры

Результаты работы методов должны формировать SQL-запросы для каждой операции темпоральной алгебры [9]. На рисунке 3. представлены примеры операций темпоральной алгебры.

На данном этапе реализуется обработчик, который преобразует входные данные в SQL-запрос. После реализации темпоральной алгебры любой темпоральнй запрос может быть транслирован в реляционный.

В целях дополнительной обработки темпоральной информации используются дополнительные методы, результатом которых будет SQL-запрос к нашей БД. К примеру, при помощи SQL-запросов имеется возможность поиска максимального дохода за определенный промежуток времени или заработной платы сотрудника в интересующее время. На рисунке 4 представлен код запроса поиска максимального дохода за месяц с использованием метода проекции из темпоральной логики.

На рисунке 5 представлен код запроса для определения заработной платы работника, с использованием метода getSalary, входными параметрами является работник компании и выбранный период времени.

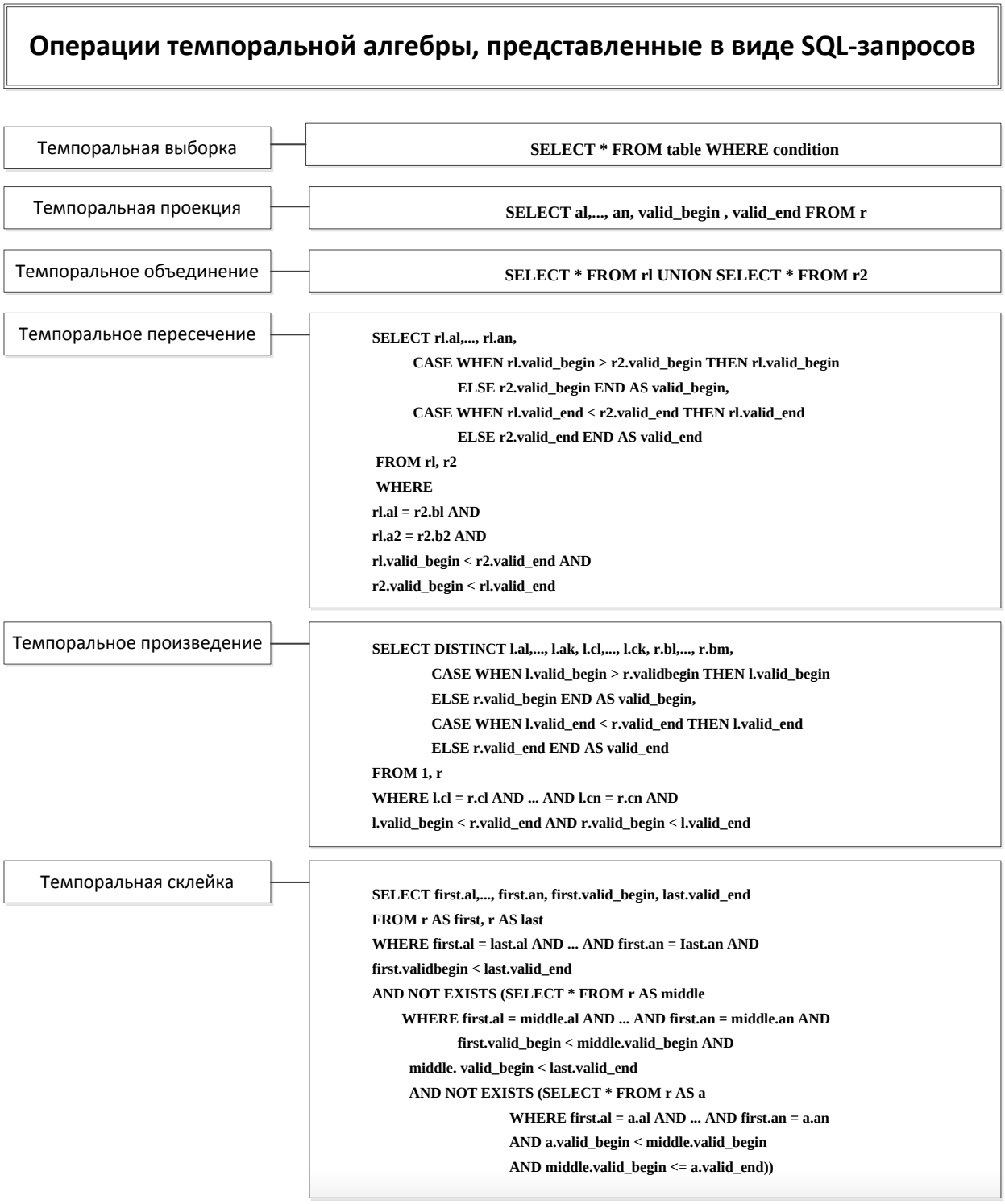

Рисунок 3. – Операции темпоральной алгебры, представленные в виде SQL-запросов

```
1       public List<Income> maxIncome(int startMonth, int endMonth) {
2          sf = HibernateUtil.getSessionFactory();
3          sf.getCurrentSession();
4          Session session = sf.openSession();
5          DetachedCriteria maxId = DetachedCriteria.forClass(Profit.class)
6                .setProjection( Projections.max("profitIncome") );
7
8          Criteria criteria = session.createCriteria(Profit.class)
9                .add(Restrictions.between("profitMonth", startMonth, endMonth))
10               .add( Property.forName("profitIncome").eq(maxId) );
11         return criteria.list();
12      }
```

Рисунок 4. – Код запроса для поиска максимального дохода за месяц

Проблема использования темпоральных данных заключается в отсутствии их эффективной поддержки в языке запросов SQL. Реляционные базы данных позволяют хранить информацию лишь о текущем состоянии реального мира, поэтому поддержка работы темпоральных данных осуществляется при помощи различных методов, реализуемых разработчиками.

```
1       public List<TempSalary> getSalary(Employee employee, Date start, Date end) {
2          sf = HibernateUtil.getSessionFactory();
3          Session session = sf.openSession();
4          DetachedCriteria maxId = DetachedCriteria.forClass(Salary.class)
5                .setProjection( Union.max("profitIncome") );
6
7          Criteria criteria = session.createCriteria(Profit.class)
8                .add(Restrictions.ge("Temp_Salary_Start", start)).add(Restrictions.lt("Temp_Salary_Start", end));
9          return criteria.list();
10
```

Рисунок 5. – Код запроса для определения заработной платы работника

В данном примере представлена реализация темпоральной базы данных для компании, осуществляющей проектную деятельность, где в целях представления времени в темпоральной модели данных использовались

атрибуты, связанные с фиксацией показаний времени [10].

Основным приемом логического моделирования темпоральных данных является добавление временных меток в сущностях предметной области. Данная операция позволяет зафиксировать в модели поведение некоторых атрибутов сущности во времени, для чего в данном примере создана диаграмма сущностей (рисунок 2), отражающая темпоральную поддержку и связи между ними. Для операции темпоральной алгебры в данной работе были рассмотрены примеры методов, транслирующих темпоральный запрос в реляционный.

Представленный в данном примере подход к моделированию темпоральных данных является расширением модели "сущность-связь". Знание техники построения темпоральных моделей данных особенно важно при проектировании баз данных, так как одной из целей их создания является исследование временных зависимостей данных.

Литература

1. Костенко Б.Б., Кузнецов С.Д., История и актуальные проблемы темпоральных баз данных // Труды ИСП РАН. 2007. № 2. URL: cyberleninka.ru/article/n/istoriya-i-aktualnye-problemy-temporalnyh-baz-dannyh.

2. Шаповалова Е. Ю. Мультииерархическая модель управления предприятием на естественном языке // Инженерный вестник Дона. – 2018. – № 2. URL: ivdon.ru/ru/magazine/archive/N2y2018/5032

3. Прищепа М. В. Математическое обеспечение распределенной системы диагностирования устройств железнодорожной автоматики и телемеханики // Инженерный вестник Дона. 2007. №. 2. URL: ivdon.ru/ru/magazine/archive/n2y2007/27

4. Allen J.F. Towards a general theory of action and time. – Artificial Intelligence, 23(2), 1984. pp 123–154.

5. McDermott D. A temporal logic for reasoning about processes and plans // Cognitive science. – 1982. – V. 6. – №. 2. – pp. 101-155.

6. Тейз А., Грибомон П. Логический подход к искусственном интеллекту. От модальной логики к логике баз данных. Пер. с франц. – М.: Мир, 1998. 494 с.

7. Дворецкий С., Таров В., Муратова Е. Информационные технологии в подготовке инженеров // Высшее образование в России. – 2001. – №. 3. – С. 130-135.

8. Нуген Д.К. Организация доступа, хранения и извлечения знаний в темпоральных базах данных: специальность 05.13.11: диссертация на соискание ученой степени кандидата технических наук / Санкт-Петербург, 2006. – 176 с.

9. Балдин А. В., Тоноян С. А., Елисеев Д. В. Язык запросов к миварному представлению реляционных баз данных, содержащих архив информации из предыдущих кадровых систем // Инженерный журнал: наука и инновации. – 2013. – №. 11 (23). – С. 20.

10. Котиков П. Е., Нечай А. А., Зацепин В. А. Темпоральные базы данных и язык запросов // Наука и современность. – 2014. – №. 2. – С. 60.

References:

1. Kostenko B.B., Kuznecov S.D., Istoriya i aktual'nye problemy temporal'nyh baz dannyh. Trudy ISP RAN. 2007. № 2. URL: cyberleninka.ru/article/n/istoriya-i-aktualnye-problemy-temporalnyh-baz-dannyh.

2. Shapovalova E. YU. Inzhenernyj vestnik Dona. 2018. №. 2. URL: ivdon.ru/ru/magazine/archive/N2y2018/5032

3. Prishchepa M. V. Inzhenernyj vestnik Dona. 2007. №. 2. URL: ivdon.ru/ru/magazine/archive/n2y2007/27

4. Allen J.F. Towards a general theory of action and time. Artificial Intelligence, 23(2), 1984. Pp. 123–154.

5. McDermott D. Cognitive science. 1982. V. 6. №. 2. pp. 101-155.

6. Tejz A., Gribomon P. Logicheskij podhod k iskusstvennom intellektu: Ot modal'noj logiki k logike baz dannyh. [A logical approach to artificial intelligence. From modal logic to database logic]. M.: Mir, 1998. P.494.

7. Dvoreckij S., Tarov V., Muratova E. Vysshee obrazovanie v Rossii. 2001. №. 3. pp. 130-135.

8. Nugen D.K. Organizaciya dostupa, hraneniya i izvlecheniya znanij v temporal'nyh bazah dannyh: special'nost' [Organization of access, storage and retrieval of knowledge in temporal databases]. Dissertaciya na soiskanie uchenoj stepeni kandidata tekhnicheskih nauk. Sankt-Peterburg, 2006. 176 p.

9. Baldin A. V., Tonoyan S. A., Eliseev D. V. Inzhenernyj zhurnal: nauka i innovacii. 2013. № 11 (23). P. 20.

10. Kotikov P. E., Nechaj A. A., Zacepin V. A. Nauka i sovremennost'. 2014. №. 2. P. 60.