**Дзюба А.Г., Щербаков С.М.**

**РГЭУ (РИНХ), г. Ростов-на-Дону, Россия**

**Анализ методов хранения исторических данных в реляционных БД**

**Аннотация:** В статье дается определение исторических данных, приводится три самых популярных метода хранения исторических данных, описываются их преимущества и недостатки. Также приводятся примеры SQL-запросов для работы с историческими данными в рамках описанных методов. И подводится итог того, в каких ситуация стоит применять тот или иной подход.

**Ключевые слова:** метод, хранение, исторические данные, изменение объекта, sql-запрос.

При разработке сложных информационных систем (ИС) нередко приходится закладывать возможность определение состояния объектов за определенный период. Это может быть связанно как с задачами отчетности (внутренней, налоговой инспекцией и т.д.), так и с задачами аналитики (выявление каких-либо закономерностей и проверке гипотез и т.д.). Поэтому вопрос хранения изменений состояния объектов ИС является важным и требующим ответа еще на этапе проектирования базы данных (БД).

Исторические данные – это данные, которые отражают не только текущее состояние объекта, но и его прошлые состояния с момента создания. Исторические данные позволяют ответить на вопросы: что изменилось в объекте, кто внес изменения и когда они были внесены [3].

Такие данные нужны для отслеживания изменений, например, статуса заказа, цены на товар и других характеристик объектов.

При рассмотрении задачи хранения исторических данных будем ориентироваться на реляционные базы данных, как средство долговременного хранения данных со строгой организацией и структурой [5].

Существует несколько основных подходов к хранению исторических данных, в рамках БД, в зависимости от задач, при этом они могут быть скомбинированы.

Метод хранения 1. Хранение версий

Например, при изменении объекта, соответствующего таблице «Заказ» в этой же таблице будет создана запись, представляющая новую версию объекта с актуальным на текущий момент состоянием [4].

На рисунке 1 представлен пример такого метода хранения исторических данных.

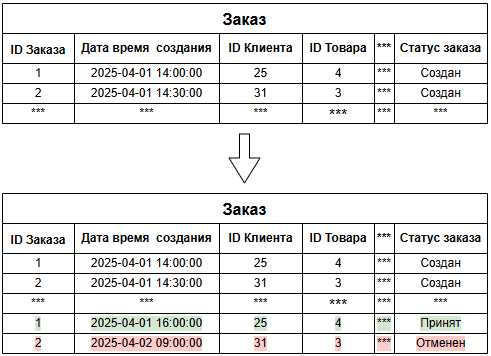


Рисунок 1 – Пример метода хранение версий объекта

Обратим внимание, что в качестве первичного ключа для идентификации записи используется составной ключ («ID Заказа» и «Дата время создания»). «ID Заказа» — определяется при создании объекта и далее остается неизменным.

Основное преимущество метода хранение версий объекта — это простота поиска состояния объекта в нужном периоде, поскольку все состояния хранятся в одной таблице.

Но при частом изменении состояния объектов использование подхода хранения версий не желательно, так как таблица быстро увеличивается в размерах, что приводит к увеличению времени выполнения SQL-запросов.

Чтобы определить актуальное состояние достаточно составить SQL-запрос так, чтобы поле «Дата время создания» было максимальным:

*SELECT [ID Заказа], MAX([Дата время создания]) FROM [Заказ] GROUP BY 1;*

Пример SQL-запроса состояния объекта за определенный период:

*SELECT \* FROM [Заказ]*

*WHERE [ID Заказа] = 1*

*AND [Дата время создания] >= '2025-04-01 09:00:00'*

*AND [Дата время создания] <= '2025-04-30 09:00:00';*

Для просмотра всех состояний конкретного объекта, от актуальной записи к самой первой можно использовать SQL-запрос:

*SELECT \* FROM [Заказ] WHERE [ID Заказа] = 1 ORDER BY [Дата время создания] DESC;*

Метод хранения 2. Журнал изменений

В отличии от предыдущего метода, данный подход при любом изменении записей в таблице «Заказ» создаются записи в таблице «Журнал изменений заказа» [2]. На рисунке 2 продемонстрирован пример этого метода.

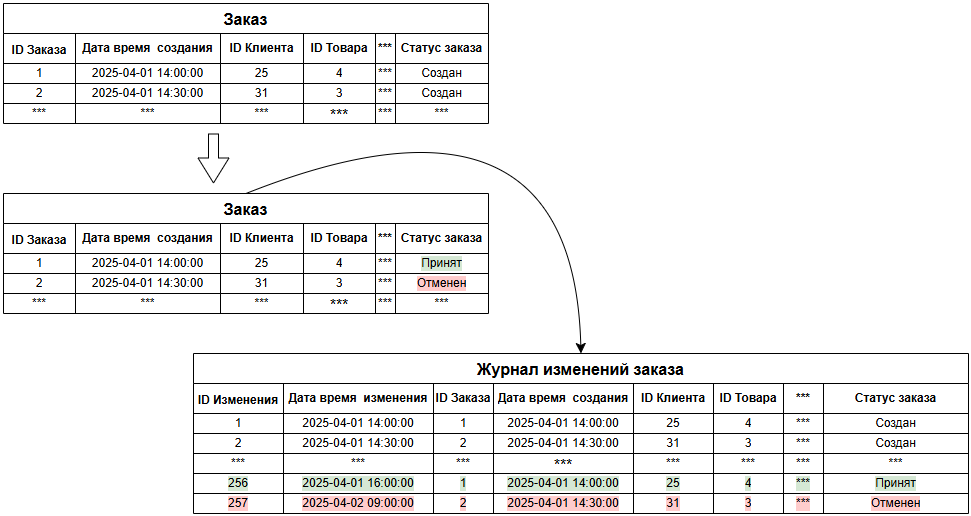


Рисунок 2 – Пример метода журнал изменений объекта

Из рисунка следует, что данные в таблице «Заказ» всегда имеют актуальное состояние, а в таблице «Журнал изменений» создается ее копия с временной меткой «Дата время изменения», что также позволяет отслеживать изменения состояния объекта во времени [2].

Главным преимуществом такого подхода является быстрый доступ к актуальному состоянию объекта, а недостаток все тот же, быстрое увеличение размеров таблицы «Журнал изменений заказа», увеличению времени выполнения SQL-запросов, использующих данные из нее.

Обычно этот подход применяется в тех случаях, когда исторические данные используются не часто.

Пример SQL-запроса актуального состояния объекта:

*SELECT \* FROM [Заказ] WHERE [ID Заказа] = 1;*

Пример SQL-запроса состояния объекта за определенный период:

*SELECT \* FROM [Заказ]*

*INNER JOIN [Журнал изменений заказа] ON [Заказ].[ID Заказа] = [Журнал изменений заказа].[ID Заказа]*

*WHERE [Заказ].[ID Заказа] = 1*

*AND [Журнал изменений заказа].[Дата время изменения] >= '2025-04-01 09:00:00'*

*AND [Журнал изменений заказа].[Дата время изменения] <= '2025-04-31 09:00:00';*

Пример SQL-запроса всех состояний конкретного объекта, от актуальной записи к самой первой:

*SELECT \* FROM [Заказ]*

*INNER JOIN [Журнал изменений заказа] ON [Заказ].[ID Заказа] = [Журнал изменений заказа].[ID Заказа]*

*WHERE [Заказ].[ID Заказа] = 1 ORDER BY [Дата время изменения] DESC;*

Метод хранения 3. Событийное хранение

Этот подход фиксирует не состояние объекта во времени, а событие произошедшие с объектом. На рисунке 3 представлен пример этого метода [1].

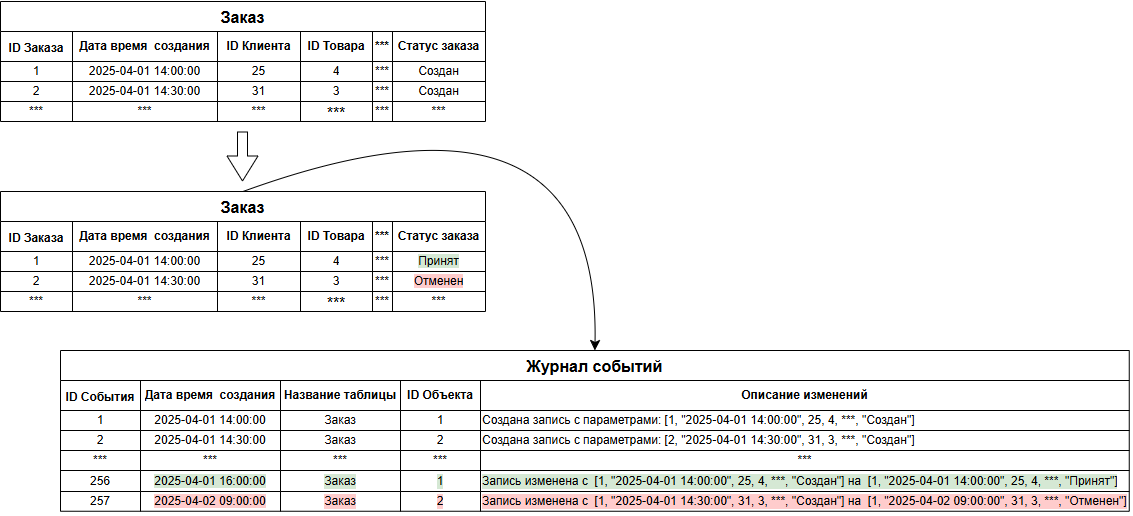


Рисунок 3 – Пример метода таблица истории

На рисунке показано, что при изменении объекта в таблице «Заказ» записи обновляются до актуального состояния, а в таблице «Журнал событий» появляются записи с описанием события произошедшего с объектом [1].

При создании, изменении или удалении объекта из основной таблицы в таблице «Журнал событий» появляется запись с временной меткой «Дата время создания» и описанием изменений в атрибуте «Описание изменений». А атрибуты «Название таблицы» и «ID объекта» используются для определения объекта в основной таблице объектов.

Благодаря этому подходу можно легко восстанавливать любое состояние объекта и отслеживать любые изменения, не боясь потери данных.

Но этот подход относительно сложно реализовать, так как не каждая система управления базами данных (СУБД) обладает необходимыми инструментами, не редко приходится прибегать к сторонним инструментам и сервисам, что увеличивает сложность разработки и поддержания БД, а соответственно накладные расходы.

Пример SQL-запроса актуального состояния объекта:

*SELECT \* FROM [Заказ] WHERE [ID Заказа] = 1;*

Пример SQL-запроса с отображением событий, произошедших с объектом за определенный период:

*SELECT [Журнал событий].\* FROM [Заказ]*

*INNER JOIN [Журнал событий] ON [Заказ].[ID Заказа] = [Журнал событий].[ID Объекта] AND [Журнал событий].[Название таблицы] = 'Заказ'*

*WHERE [Заказ].[ID Заказа] = 1*

*AND [Журнал событий].[Дата время создания] >= '2025-04-01 09:00:00'*

*AND [Журнал событий].[Дата время создания] <= '2025-04-31 09:00:00';*

Пример SQL-запроса всех событий, произошедших с конкретным объектом, от актуальной записи к самой первой:

*SELECT [Журнал событий].\* FROM [Заказ]*

*INNER JOIN [Журнал событий] ON [Заказ].[ID Заказа] = [Журнал событий].[ID Объекта] AND [Журнал событий].[Название таблицы] = 'Заказ'*

*WHERE [Заказ].[ID Заказа] = 1*

*ORDER BY [Журнал событий].[Дата время создания] DESC;*

На сегодняшний день еще не придуман идеальный метод работы с историческими данными. Каждый из ранее представленных подходов имеет свои преимущества и недостатки. Чтобы понять какого из представленных подходов стоит придерживаться следует ответить на вопросы в таблице 1.

Таблица 1 – Вопросы для определения нужного метода хранения исторических данных.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Метод хранения 1. Хранение версий | Метод хранения 2. Журнал изменений | Метод хранения 3. Событийное хранение |
| Как много данных хранит таблица? | Мало | Много | Много |
| Как часто в ней происходят изменения? | Редко | Среднее | Часто |
| Как часто нужен доступ к историческим данным? | Редко | Часто | Часто |
| Удаляются ли данные из таблицы? | Не удаляются | Не удаляются | Удаляются |
| Сколько ресурсов потребуется выделить на реализацию того или иного подхода? | Минимум | Среднее | Много |

Если при проектировании БД были даны ответы на вопросы, то сложится четкое понимание, какой из методов стоит использовать в том или ином случае для хранения и работы с историческими данными. Затраченные ресурсы на анализ предметной области в части анализа работы с историческими данными в последствии помогут сэкономить много больше времени и ресурсов (направленных на правки, доработку или же разработку более подходящих решений).

Список литературы

1. Петров А. Распределенные данные. Алгоритмы работы современных систем хранения информации / А. Петров. – СПб.: Питер, 2021. – 336с.
2. Котельников А.С., Якунина В.Н., Рысина А.Д. Сбор и обработка исторических данных в автоматизированных информационных системах / А.С. Котельников, В.Н. Якунина, А.Д. Рысина // Прикладная информатика – 2012. – №6. – С.9-14.
3. Малакшинов, С. Версионность и история данных. [Электронный ресурс] / С. Малакшинов // Habr. – 2010. – Режим доступа <https://habr.com/ru/articles/101544/> .
4. StackOverFlow. Хранение истории изменений в единой таблице. [Электронный ресурс] / StackOverFlow // StackOverFlow – 2018. – Режим доступа <https://ru.stackoverflow.com/questions/847191/> .
5. Андреев Р. Виды баз данных. [Электронный ресурс] / Р. Андреев // Академия Selectel – 2022. – Режим доступа <https://selectel.ru/blog/databases-types/> .

ФИО: Дзюба Андрей Григорьевич

Номер группы: ЭК-911

e-mail: adzyuba1997@yandex.ru

телефон: +7(933)931-94-09