Пилецкий И.И., Селицкий Д. В., Костиков В.А.

JV IBA, г. Минск, РБ

ipiletski@iba.by, +375 17 2256578

Проблемы и решения построения реляционных СУБД с историей вносимых изменений

В докладе на примере разработки конкретной базы данных (БД), приводятся четыре различных подхода к построению моделей БД, их соответствию некоторым общепринятым критериям, и дается оценка соответствия данных моделей выбранным критериям.

Моделирование выполнено на языке моделирования IDEF1x.

Реализация выполнена в среде СУБД DB2 UDB.

In this article based on the example of the development of the concrete data base (BD), are given four different approaches to the construction of models BD, their correspondence to some conventional criteria, and evaluation of the correspondence of data of models to the selected criteria is given

1. Основные проблемы

не простые, а иногда и противоречивые требования к БД

- БД нормализованная или слабо нормализованная;
- Архитектура и структура гибкая и модифицируемая или заранее определенная (жесткая);
- Не деградирует в процессе эксплуатации или деградирует и при определенном уровне деградации допускается реорганизация БД;
- Не сложное ПО для обслуживания БД и жесткая структура или сложное ПО для обслуживания гибкой, изменяющейся структуры БД;
- БД обеспечивает достоверность и не противоречивость хранимых данных, но и одновременно слабо нормализованная;
- Имеет несколько разных источников пополнения информации или один;
- Идентификация записей внутренняя или внешняя
- и т.д..
- БД отражает историю хранимых данных или нет

2. Историчность

- **Суть проблемы историчности хранимых данных можно определить как: обеспечить длительное хранение и доступ к информации с историей ее изменения на протяжении всего времени нахождения в БД, и после окончания «жизни» объекта.**

- **В таких БД доступ к данным (в принципе) такой же, как и в реляционных, но механизм корректировки данных другой. Общепринятые операции по редактированию данных (вставка, корректировка, удаление) не могут выполняться средствами РБД над конкретной текущей записью. Данные, содержащиеся в текущей записи, нельзя изменять, они должны храниться длительное время, они должны содержать ту информацию, которая у них есть, которая была актуальной на определенный период. Удаление записи приводит к установлению даты конца «жизни» объекта у родителя со всеми потомками.**

- **Каждый объект БД и каждая запись объекта имеет специальные атрибуты, которые обеспечивают уникальность (для одного внешнего объекта может быть несколько записей) и историчность хранимых данных (внешний идентификатор + дата совершения операции или суррогатный идентификатор, период (даты начала и конца) «жизни» объекта – который делиться при изменении характеристик объекта).**

3. Основные требования

- **основные требования**:
- – быть гибкой и расширяемой в процессе сопровождения*,
- – быть оптимальной по времени поиска данных,
- – не деградировать в процессе длительной эксплуатации,
- – обеспечивать достоверность и не противоречивость хранимых данных (в том числе и исторических),
- – содержать все данные о корректировке за длительный промежуток времени, т.е. быть исторической,
- – поддерживать независимый интерфейс доступа к БД,
- - поддерживать разработку несложного заказного обслуживающего ПО
- ***Замечание. В силу специфики предметной области невозможно заранее определить все необходимые сущности и их атрибуты для построения БД. В процессе эксплуатации БД должна развиваться и содержимое сущностей должно уточняться.**

4. Методология построения БД

- **Вся БД НСИ БелЖД построена на основании выделения логических сущностей (сейчас более 22), таких как «дорога», «станция», «поезд», «локомотив», «вагон», «контейнер», «груз», «клиентура», «тариф», «клиент», «тарифное руководство N 4», «план формирования поездов», «технологические таблицы» и т.д.**
- **В данной работе, все результаты приведены для реальных подсхем (Логических сущностей) «Дорога» и «Станции» стран СНГ. Подсхема «Станция» связана с подсхемой «Дорога».**
- **Л.С. «Дорога» –**

  «Регион» «Страна» «Администрация» «Дорога» «Отделение дороги»
- **Л.С. «Станция» –**

  «Станция» «Справочник межгосударственных стыков» «Справочник стыков» «Признак стыка» «Параграф станции» «Станции и их признаки» «Пограничный переход» «Признаки станций» «Выделенные станции» «Ближайшие выделенные станции» «Расстояние и время хода от выделенной до невыделенной станции» «Таблица соответствия станций параграфам» «Характеристика станции» «Порядок выдачи станций в отчётах» и др.

- **Варианты идентификации объектов – естественные и суррогатные ключи**

4.1. Модель с естественной идентификацией объектов

- **Уникальность записи -**

уникальный кода объекта + дата вставки записи (PK = код и даты вставки объекта).

БД содержит не только актуальные записи в данный момент, но и исторические.

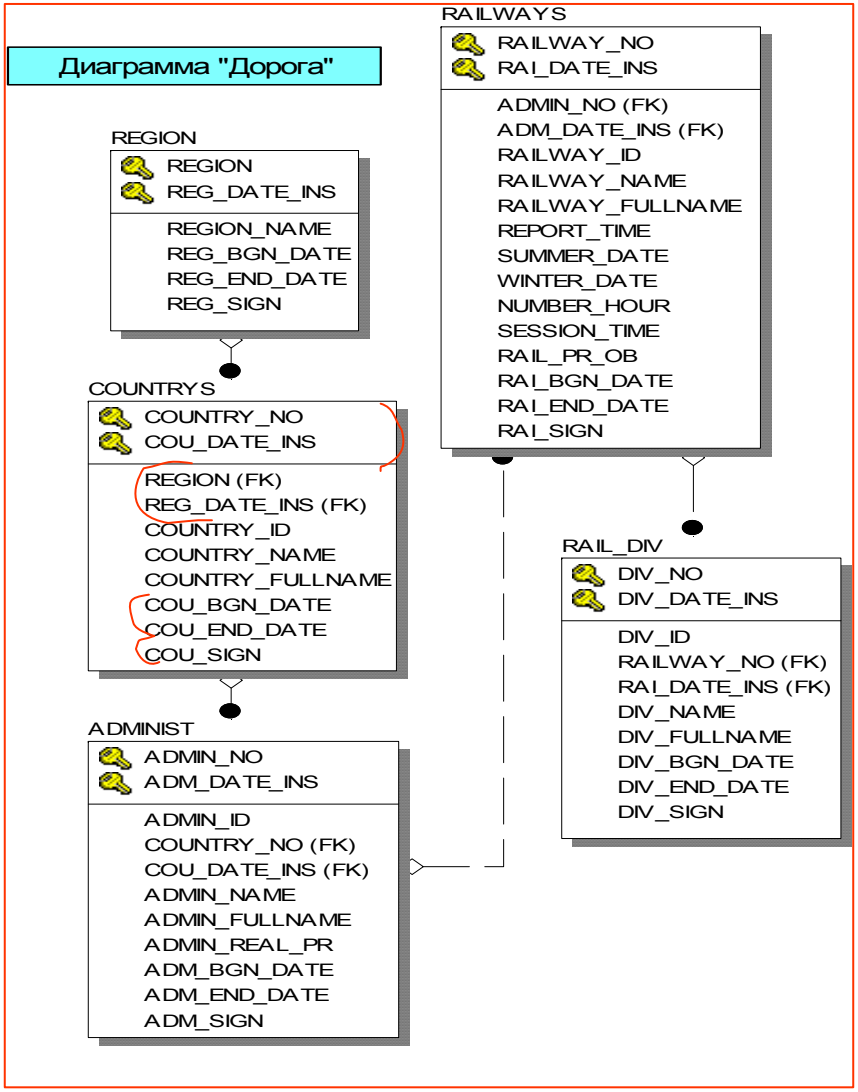
Поэтому в таблицах допускается наличие нескольких записей для одного и того же объекта.

Историческая принадлежность записи к родителю, по ссылке, устанавливается на основании диапазона жизни объекта и значения кода объекта.

Физически исторические записи выведены из схемы (PK-FK, в FK остается только код объекта), что является не совсем удобным при работе с ними.

- **В данной модели совмещены принципы реляционности для активных записей (как для любой реляционной БД) и искусственного введения в нее историчности.**

Ко всем записям (в том числе и историческим) обеспечивается доступ по ключам трех типов (PK, AK, IE).



4.1. Часть 2

- **PARAGRAF—STATION_PARAGRAF –-STATION**
  (многие - ко - многим)

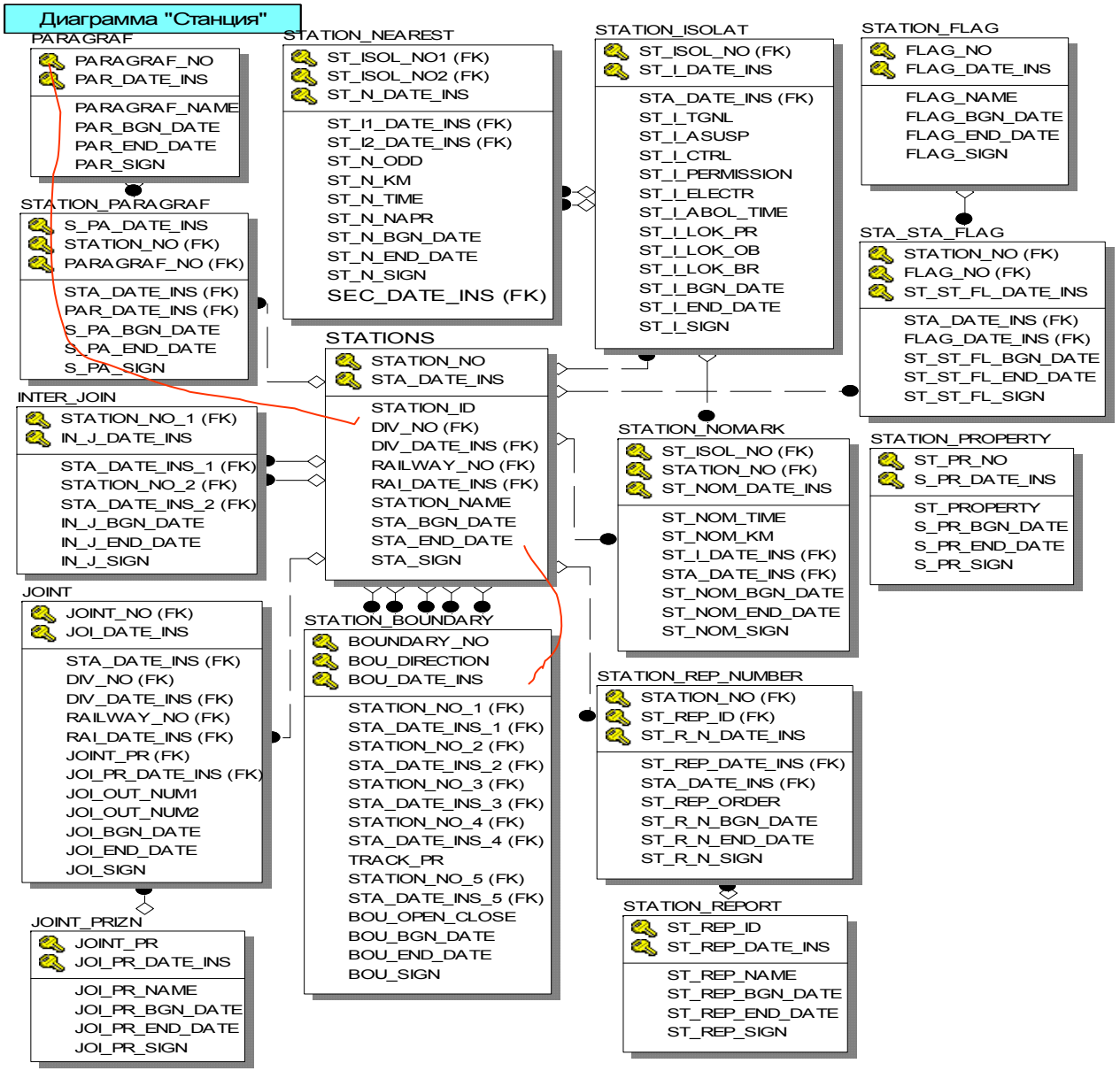
STATION – STATION_BOUNDARY
(состоит из 5- станций)

Данная концепция позволяет:

- минимизировать объем используемой дисковой памяти для хранения данных,

- обеспечить быстрый доступ к записям актуальным в настоящее время,

- обеспечить доступ ко всем историческим записям, если это необходимо.



4.2. Модель с естественной идентификацией объектов и нумерацией исторических записей

- **Этот метод построения модели базы данных является частичной модификацией первого.**

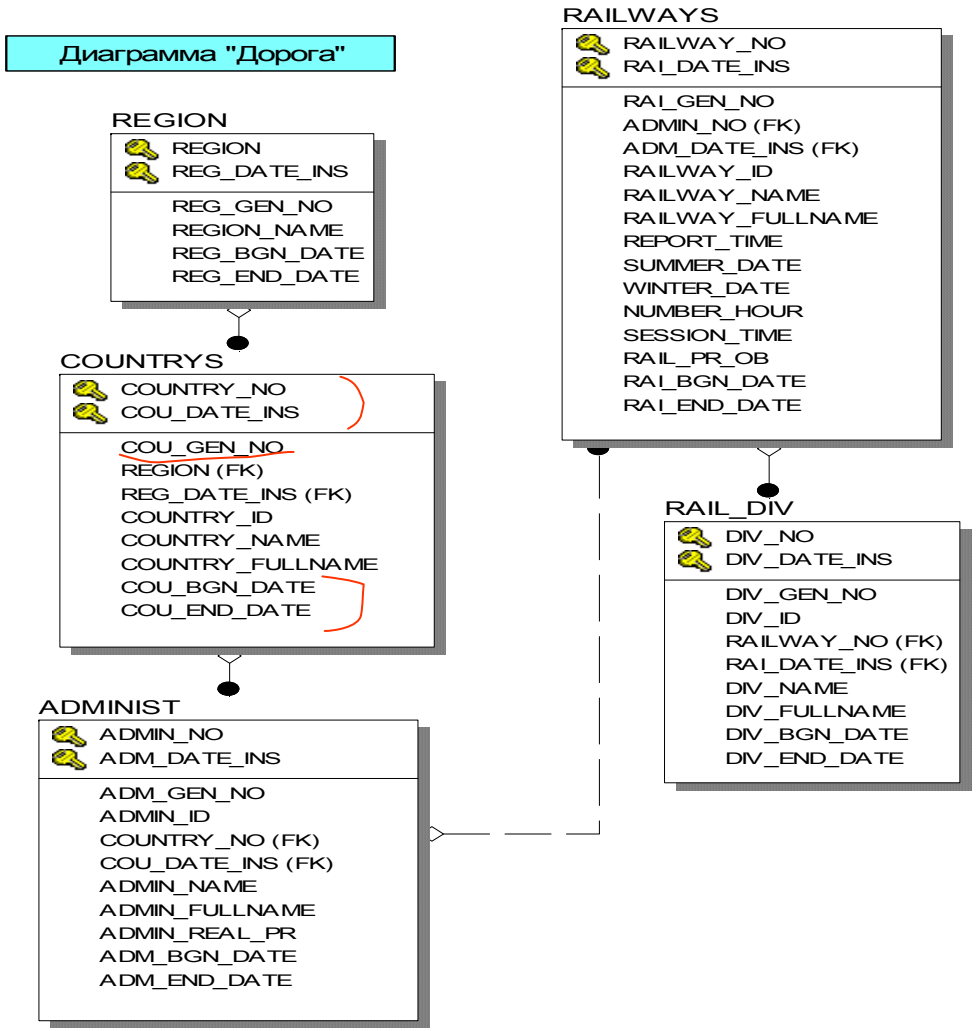
В данном методе вводится дополнительный атрибут, указывающий номер поколения записи.

Приведенная модель похожа на предыдущую. В ней также совмещены принципы реляционности для актуальных записей и искусственное введение исторических записей.

Исторические записи сознательно выводятся из схемы и хранятся вне схемы, но в отличие от предыдущего варианта в данной модели исторические записи, относящиеся к одному объекту, явно связаны друг с другом с помощью номера поколения записи, что приводит к их упорядоченности.

К записям осуществляется доступ по ключам двух видов РК и АК и по номеру поколения.

- **Данный метод, как и предыдущий, не отражает прямых связей исторических записей с их родительскими записями.**

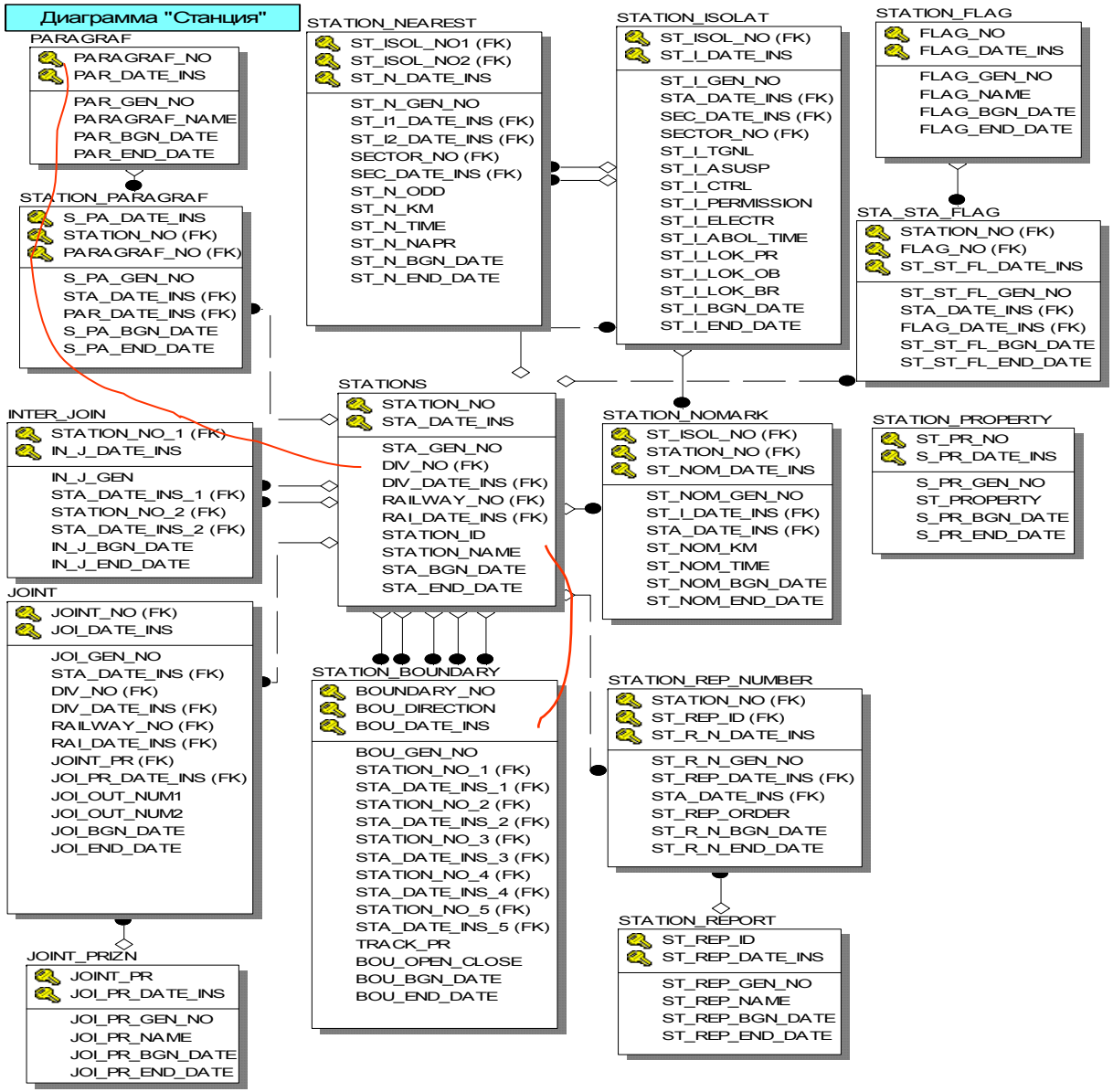


4.2. Часть 2

- **PARAGRAF – STATION_PARAGRAF – STATIONS (многие - ко - многим)**

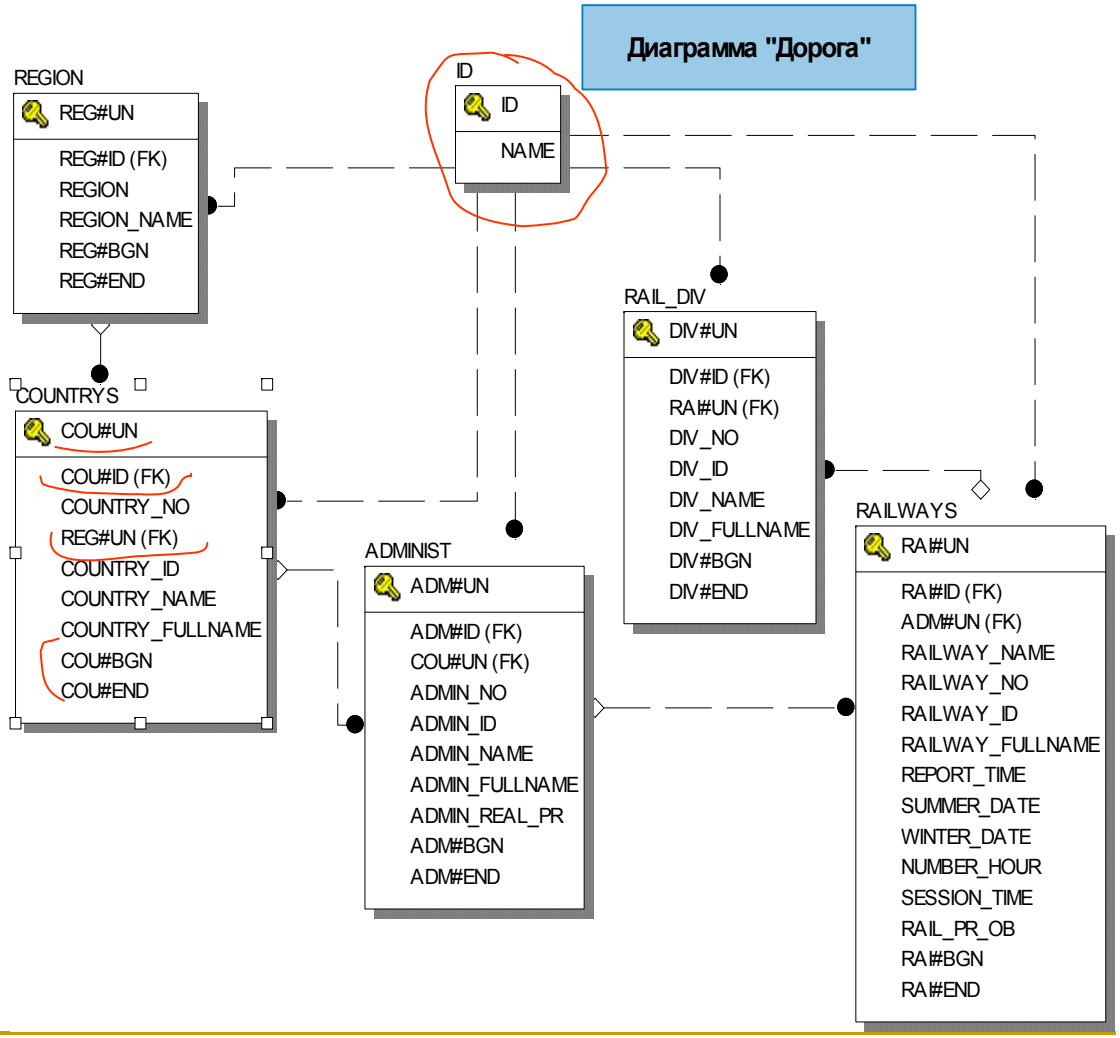
STATIONS – STATION_BOUNDARY (5 – станций определяют пограничный - переход)

- **Этот метод позволяет минимизировать объем дисковой памяти для хранения данных, и обеспечить быстрый доступ к активным записям актуальным в настоящее время и по номеру поколения ко всем историческим.**



4.3. Модель с искусственной идентификацией объектов и сохранением дерева историчности объектов

- В этой модели БД используется суррогатная идентификация объектов, которая позволяет решить проблему идентификации исторических записей за счет введения внутреннего идентификатора (ID) – объекта и выработки внутреннего суррогатного первичного ключа (PK) для каждой записи в БД.
- В результате изменения характеристик хранимых данных происходит дублирование поддеревьев-потомков для корректируемых записей, но данный метод обеспечивает прямые связи с исторических записей с их родительскими записями.
- Концепция построения БД приводит к тому, что РБД, представленная плоской схемой, превращается в многомерную «пирамидальную» БД, в которой нарушены основные принципы построения нормализации реляционных БД.



4.3. Часть 2.

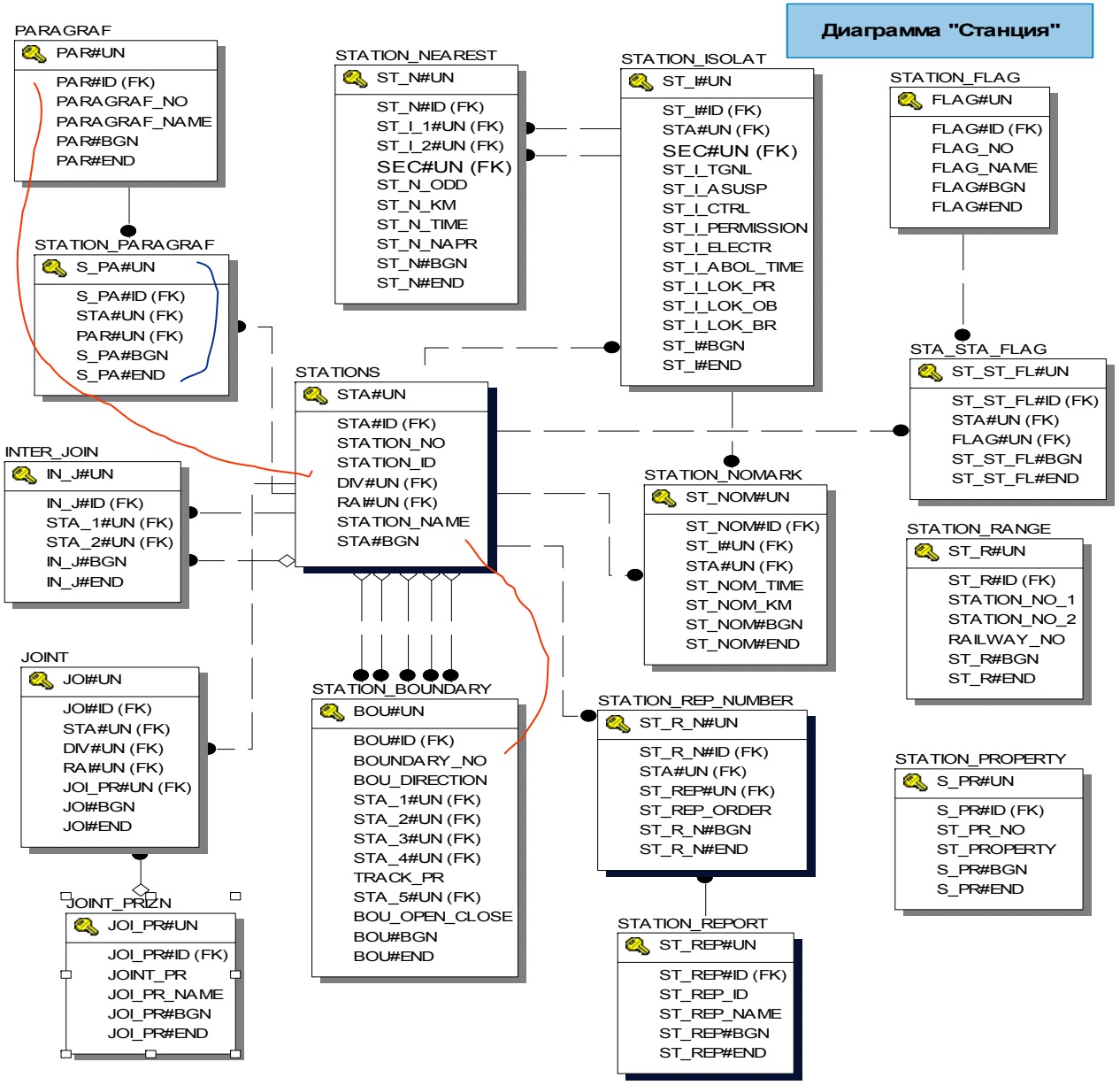
- Много служебных полей
- Записи таблиц «многие – ко многим» полностью состоят из служебных полей

Но данный метод обеспечивает прямые связи с исторических записей с их родительскими записями.

Временные срезы – нормализованные.

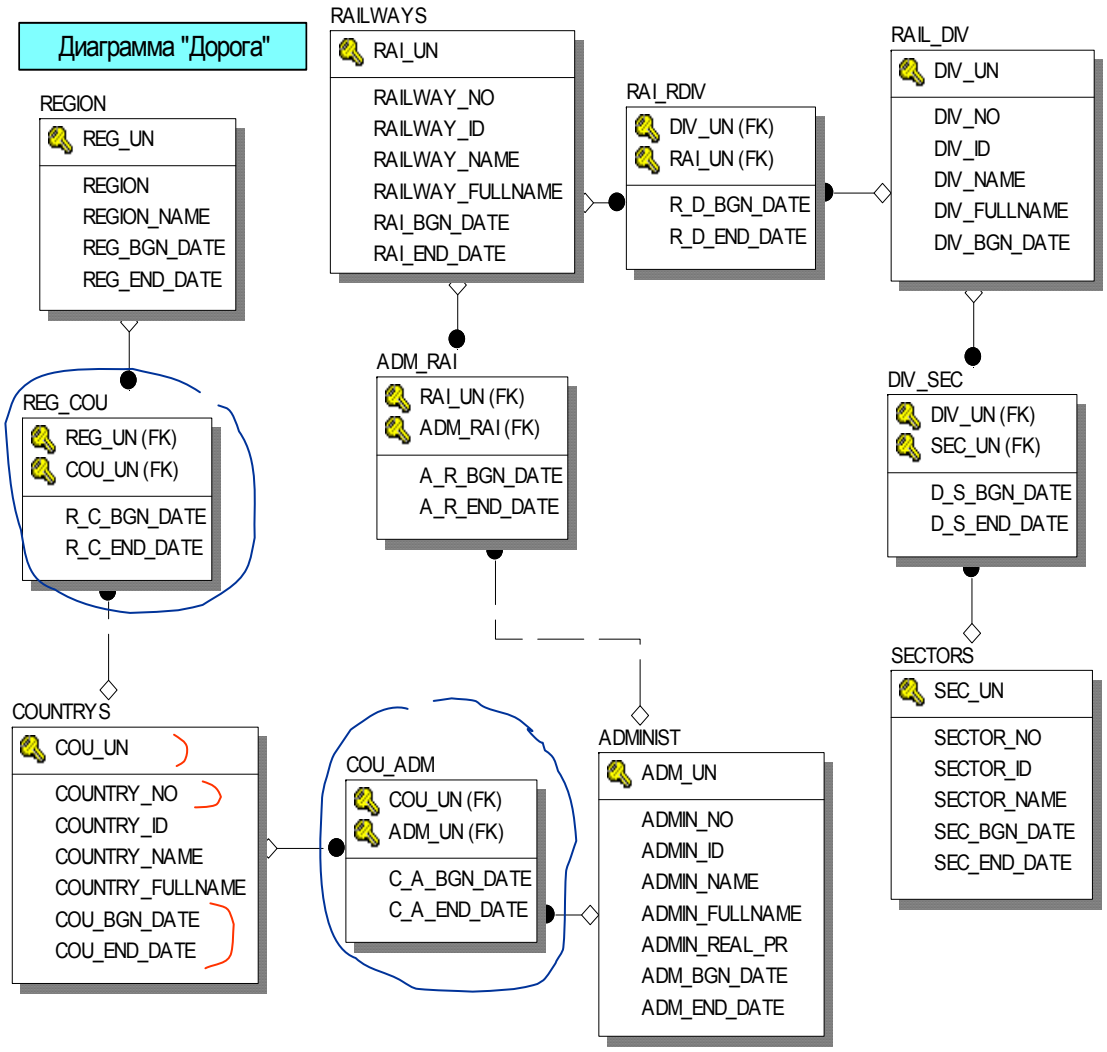
Резкий рост хранимых данных

Сложное ПО.

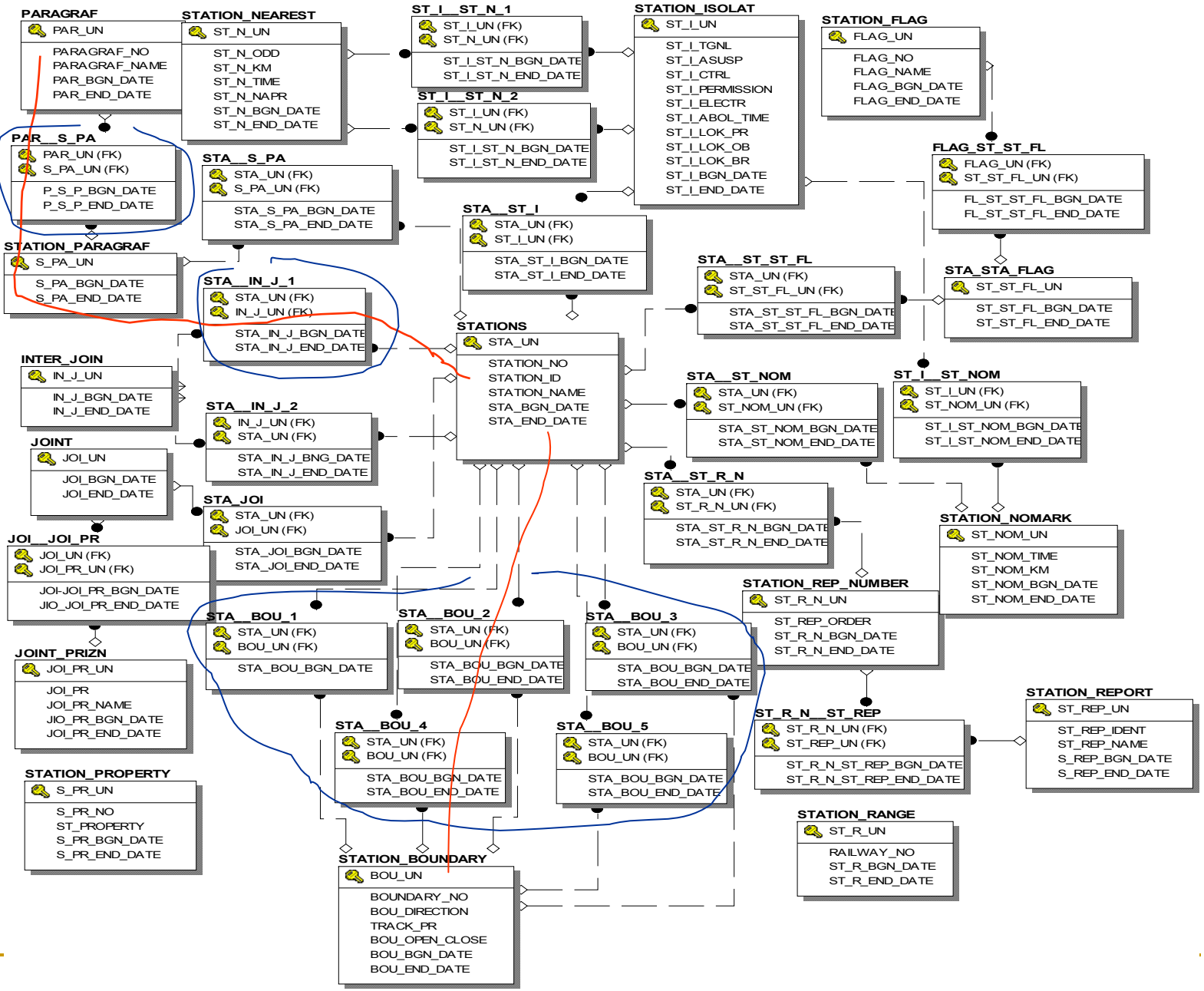


4.4. Модель с искусственной идентификацией объектов и с таблицами переходниками

- Этот метод построения модели БД основан на введении в схему БД таблиц-переходников, которые помогают разрешить связи “многие ко многим”, возникающие между родителями и потомками вследствие историчности БД, и избежать построения многомерных деревьев в БД.
- В таблицах данных хранятся актуальные и исторические записи, а в таблицах-переходниках разрешаются связи
- В схеме БД для разрешения связей «многие ко многим», вместо построения поддерева исторических записей, для каждого отношения строится таблица - переходник. Первоначально мощность множеств записей данных таблиц можно оценить как, декартово произведение множеств записей исходных таблиц, участвующих в отношениях.
- Данный подход к построению РБД по сравнению с предыдущей моделью прячет суть проблемы – дублирование поддеревьев, в таблицах-переходниках и усложняет модель БД.



4.4.


PARAGRAF
PAR_UN
PARAGRAF_NO
PARAGRAF_NAME
PAR_BGN_DATE
PAR_END_DATE
STATION_NEAREST
ST_N_UN
ST_N_ODD
ST_N_KM
ST_N_TIME
ST_N_NAPR
ST_N_BGN_DATE
ST_N_END_DATE
ST_I__ST_N_1
ST_I_UN (FK)
ST_N_UN (FK)
ST_I_ST_N_BGN_DATE
ST_I_ST_N_END_DATE
ST_I__ST_N_2
ST_I_UN (FK)
ST_N_UN (FK)
ST_I_ST_N_BGN_DATE
ST_I_ST_N_END_DATE
STATION_ISOLAT
ST_I_UN
ST_I_TGNL
ST_I_ASUSP
ST_I_CTRL
ST_I_PERMISSION
ST_I_ELECTR
ST_I_ABOL_TIME
ST_I_LOK_PR
ST_I_LOK_OB
ST_I_LOK_BR
ST_I_BGN_DATE
ST_I_END_DATE
STATION_FLAG
FLAG_UN
FLAG_NO
FLAG_NAME
FLAG_BGN_DATE
FLAG_END_DATE
FLAG_ST_ST_FL
FLAG_UN (FK)
ST_ST_FL_UN (FK)
FL_ST_ST_FL_BGN_DATE
FL_ST_ST_FL_END_DATE
PAR__S_PA
PAR_UN (FK)
S_PA_UN (FK)
P_S_P_BGN_DATE
P_S_P_END_DATE
STA__S_PA
STA_UN (FK)
S_PA_UN (FK)
STA_S_PA_BGN_DATE
STA_S_PA_END_DATE
STA__ST_I
STA_UN (FK)
ST_I_UN (FK)
STA_ST_I_BGN_DATE
STA_ST_I_END_DATE
STATION_PARAGRAF
S_PA_UN
S_PA_BGN_DATE
S_PA_END_DATE
STA__IN_J_1
STA_UN (FK)
IN_J_UN (FK)
STA_IN_J_BGN_DATE
STA_IN_J_END_DATE
STA__ST_ST_FL
STA_UN (FK)
ST_ST_FL_UN (FK)
STA_ST_ST_FL_BGN_DATE
STA_ST_ST_FL_END_DATE
STA_STA_FLAG
ST_ST_FL_UN
ST_ST_FL_BGN_DATE
ST_ST_FL_END_DATE
STATIONS
STA_UN
STATION_NO
STATION_ID
STATION_NAME
STA_BGN_DATE
STA_END_DATE
INTER_JOIN
IN_J_UN
IN_J_BGN_DATE
IN_J_END_DATE
STA__IN_J_2
IN_J_UN (FK)
STA_UN (FK)
STA_IN_J_BNG_DATE
STA_IN_J_END_DATE
STA__ST_NOM
STA_UN (FK)
ST_NOM_UN (FK)
STA_ST_NOM_BGN_DATE
STA_ST_NOM_END_DATE
ST_I__ST_NOM
ST_I_UN (FK)
ST_NOM_UN (FK)
ST_I_ST_NOM_BGN_DATE
ST_I_ST_NOM_END_DATE
JOINT
JOI_UN
JOI_BGN_DATE
JOI_END_DATE
STA_JOI
STA_UN (FK)
JOI_UN (FK)
STA_JOI_BGN_DATE
STA_JOI_END_DATE
STA__ST_R_N
STA_UN (FK)
ST_R_N_UN (FK)
STA_ST_R_N_BGN_DATE
STA_ST_R_N_END_DATE
STATION_NOMARK
ST_NOM_UN
ST_NOM_TIME
ST_NOM_KM
ST_NOM_BGN_DATE
ST_NOM_END_DATE
JOI__JOI_PR
JOI_UN (FK)
JOI_PR_UN (FK)
JOI-JOI_PR_BGN_DATE
JIO_JOI_PR_END_DATE
STA__BOU_1
STA_UN (FK)
BOU_UN (FK)
STA_BOU_BGN_DATE
STA__BOU_2
STA_UN (FK)
BOU_UN (FK)
STA_BOU_BGN_DATE
STA_BOU_END_DATE
STA__BOU_3
STA_UN (FK)
BOU_UN (FK)
STA_BOU_BGN_DATE
STA_BOU_END_DATE
STATION_REP_NUMBER
ST_R_N_UN
ST_REP_ORDER
ST_R_N_BGN_DATE
ST_R_N_END_DATE
JOINT_PRIZN
JOI_PR_UN
JOI_PR
JOI_PR_NAME
JIO_PR_BGN_DATE
JOI_PR_END_DATE
STA__BOU_4
STA_UN (FK)
BOU_UN (FK)
STA_BOU_BGN_DATE
STA_BOU_END_DATE
STA__BOU_5
STA_UN (FK)
BOU_UN (FK)
STA_BOU_BGN_DATE
STA_BOU_END_DATE
ST_R_N__ST_REP
ST_R_N_UN (FK)
ST_REP_UN (FK)
ST_R_N_ST_REP_BGN_DATE
ST_R_N_ST_REP_END_DATE
STATION_REPORT
ST_REP_UN
ST_REP_IDENT
ST_REP_NAME
S_REP_BGN_DATE
S_REP_END_DATE
STATION_PROPERTY
S_PR_UN
S_PR_NO
ST_PROPERTY
S_PR_BGN_DATE
S_PR_END_DATE
STATION_BOUNDARY
BOU_UN
BOUNDARY_NO
BOU_DIRECTION
TRACK_PR
BOU_OPEN_CLOSE
BOU_BGN_DATE
BOU_END_DATE
STATION_RANGE
ST_R_UN
RAILWAY_NO
ST_R_BGN_DATE
ST_R_END_DATE

5. Сравнительный анализ результатов моделирования

- **Для оценки результатов моделирования использовались аналитические методы оценки результатов (1- отлично – 5 - плохо) и методы, которые позволяют получить количественные результаты стоимости выполнения типичных запросов пользователей к информации хранящейся в БД**
- **5.1. Аналитические методы**
- 1. быть **гибкой и расширяемой** в процессе сопровождения, данный критерий отражает требования не только к структуре БД, но и к ПО. Но чем **проще схема БД**, тем **проще она для обслуживания ПО**;
- 2. **оптимальной по времени поиска данных**, сложность схемы БД напрямую связана с ее реактивностью;
- 3. **не деградировать в процессе длительной эксплуатации**;
- 4. обеспечивать **достоверность и непротиворечивость** данных;
- 5. **быть исторической**. Но без **резкого увеличению объема хранимой информации** при эксплуатации БД;
- 6. поддерживать **простой механизм передачи данных** (репликации данных) в другие БД. Сложная структура БД препятствует простому механизму передачи данных в другие БД (как по сложности кода, так и по времени выполнения его);
- 7. **содержать минимально необходимое количество вспомогательной технической и технологической информации**. Наличие большого количества чисто технологических атрибутов и таблиц отражает сложность построения самой БД, приводит к **разработке сложного специального заказного ПО** для обслуживания технологических атрибутов и таблиц в процессе эксплуатации;
- 8. поддерживать **независимый интерфейс** с ПО доступа к БД. Чем **проще архитектура БД**, тем проще ПО, которое используется для поиска и отображения данных.

5.2. Количественные испытания

- Метод получения количественных результатов основан на эффективности выполнения транзакций СУБД DB2 UDB 7.1 различными четырьмя физическими БД типичных тестов для пользователей БелЖД в среде Windows 2000 на компьютере P-III.
- Страны, Администрации, Дороги на территории СНГ. Выдать список всех стран, администраций и дорог, находящихся в настоящее время на территории СНГ. Входные данные для этого теста содержатся в таблицах COUNTRYS, ADMINIST, RAILWAYS и REGION.
- 2. Станции БелЖД без параграфов. Выдать все станции Белоруской железной дороги открытые для грузовой работы и работающие без параграфов. Входные данные для этого теста находятся в таблицах STATIONS, RAIL_DIV и STATION_PARAGRAF.
- 3. Справочник станций БелЖД с параграфами. Получить список всех станций Белоруской железной дороги вместе с параграфами их работы. Входные данные для запросов данного типа находятся в таблицах STATIONS, STATION_PARAGRAF и RAIL_DIV.
- 4. Список пограничных переходов БелЖД. Пограничные переходы БелЖД определяются пятью станциями: станция передачи вагонов БелЖД, стыковая пограничная станция БелЖД, стыковая пограничная станция не БелЖД, станция передачи вагонов не БелЖД и стыковой пункт учета поездов и вагонов. Входные данные для запросов данного типа содержатся в таблицах STATIONS, STATION_BOUNDARY, RAILWAYS, ADMINIST и INTER_JOIN.
- 5. Справочник стыковых пунктов по отделениям. Выдать справочник стыковых пунктов по отделениям. Входные данные для реализации данного теста находятся в таблицах JOINT, STATIONS и RAIL_DIV.
- 6. Станции, измененные в 2004-2005 г.г. Выдать список станций, претерпевших изменения в 2004-2005 гг. Входные данные для реализации данного теста находятся в таблицах STATIONS, STATION_PARAGRAF и RAILWAYS.

5.3. Результаты испытаний

Критерий оценки и тип теста	Модель 1 Оценка/ результат	Модель 2 Оценка/ результат	Модель 3 Оценка/ результат	Модель 4 Оценка/ результат
1. гибкость, расширяемость	1	1	3	4
2.Оптимальность, простота схемы БД	1	1,5	3	4\5
3. Деградируемость	1	1	4	3
4. Достоверность, непротиворечивость нормализация	1	1	3	2
5. Историчность	2,5	2	1	2
6. Сложность ПО для репликации	2	2	4	4\5
7. Сложность ПО и структуры БД	1	1	4	4\5
8. Архитектура БД – интерфейс	2	2	5	4
Тест 1.	184,59	204,46	301,44	1017,45
Тест 2.	2098,49	2308,34	2987,05	11457,84
Тест 3.	4296,35	4725,98	5687,14	23458,27
Тест 4.	2448,94	2693,83	3285,69	13371,31
Тест 5.	1039,65	1151,52	1390,92	5730,47
Тест 6.	6327,01	7007,79	9247,59	34873,85

5.4. Выводы

- В первых двух моделях хуже решается проблема доступа к историческим записям, чем в двух последующих, но обеспечивается быстрый доступ к актуальным записям и минимизируется объем хранимых данных. Первая модель БД является наиболее простой по построению запросов к актуальным записям и наиболее оптимальной по времени доступа к ним. Вторая модель незначительно уступает предыдущей по времени поиска хранимой информации.

- Модель с сохранением дерева историчности объектов не позволяет строить нормализованные БД, что приводит к избыточному хранению огромного количества пользовательских данных. Доступ к хранимым данным требует значительно больших ресурсов, чем в двух предыдущих моделях, но данная модель удобна для поддержания хранилищ данных.

- Модель с таблицами-переходниками является наихудшим вариантом для выбора ее прототипом для создания на ее основе БД. Она не оптимальна по времени поиска хранимых данных и имеет очень сложную структуру с огромным количеством служебных таблиц-переходников.