Часть 1. Пригород

Файл prog\_log (программа svod.prig\_obog\_poln )

Производит расчёт всех пригородных отчётов. Последовательный запуск в цикле программ:

svod.prig\_obog\_0(load\_); - выбор порции чтения

svod.prig\_obog\_1(load\_); - первичное чтение из пригородной базы

svod.prig\_obog\_1p(load\_) ; - первичное чтение из пассажирской базы

svod.prig\_obog\_21(load\_); -начало обработки маршрута

svod.prig\_obog\_22(load\_); - конец обработки маршрута

svod.prig\_obog\_3(load\_); - расчёт всех прочих справочников

svod.prig\_obog\_4(load\_) ; - загрузка итогов по пригороду в основную и справочные таблицы. После этого ввод реестра льготников

svod.prig\_agr\_analit(load\_); -- процедура загрузки аналитической таблицы из итогов

svod.prig\_co22(load\_); - Процедура расчёта ЦО-22 по пригороду (ещё не доделана)

Эти программы сформированы в файлах prog\_log\_2\_0, prog\_log\_2\_1, prog\_log\_2\_1p, prog\_log\_2\_2\_1, prog\_log\_2\_2\_2, prog\_log\_2\_3, prog\_log\_2\_4, prog\_log\_3\_3\_analit, prog\_log\_3\_4\_co22.

1. Программа svod.prig\_obog\_0

К началу выполнения программы уже известно, какие в принципе данные имеются в базах данных – за какие даты, с какими интервалами значений поля idnum, и в каком объёме по каждой дате.

Сперва - программа смотрит в таблице prig\_times, какие есть данные в пригородной базе, выбирает самую раннюю из порций, и если эти данные из таблицы пригородных данных, а не из пассажирской базы, и количество строк в порции свыше 400000 строк, то порция разбивается на 3 порции – первые 300000 строк, последующие максимум 300000 строк (если таковые наберутся), и всё что осталось (возможно что ничего не осталось).

Затем первая из получившихся порций получает признак направлено на обработку.

Если при следующем запуске очередная порция снова свыше 400000 строк – она тоже сперва подразбивается на 3 части.

1. Программа svod.prig\_obog\_1

Программа выполняет первый этап чтения из базы данных пригородных перевозок. И начало обработки этих данных. Она состоит из следующих основных блоков (внутри SQL-процедуры):

Rrr, Rrr0 – блок наработки целых чисел от 1 до 10000, получаемых методом рекурсивной процедуры. Тут проявляется такое ограничение среды программирования – рекурсивная процедура в SQL-запросе может быть только в самом первом блоке, соответственно лишь 1 раз за всю программу.

Prig\_main – чтение порции данных из таблицы 2 уровня l2\_prig\_main, строго за указанную дату и нужный интервал значений поля IDNUM, с выборкой всех необходимых далее поле из этой таблицы.

Prig\_cost – аналогичное чтение исходных данных из таблицы l2\_prig\_cost

Prig\_fio – аналогиная выборка из таблицы l2\_prig\_adi

prig\_main\_ - склейка главной таблицы с таблицей фамилий

prig\_cost2 – склейка prig\_cost с главной таблицей, подтягивание в неё полей полных сумм денег по маршруту.

prig\_cost2\_kbil\_nod – в теории поиск НОД (наибольшего общего делителя) сумм всех денег по каждому перегону маршрута, и полного числа билетов. На практике – определяет у каких билетов хоть в одной строке хоть одна сумма некратна числу билетов, и для них поле формирует количество групп билетов k\_bil=1. Для всех остальных далее будет k\_bil=kol\_bil

prig\_cost2\_kbil – в таблицу prig\_cost2 вставляется созданное поле k\_bil, если оно пусто – то берётся значение kol\_bil, после чего все суммы денег делятся на получившееся значение. В результате получается разбиение сумм денег на 1 группу билетов (то есть в идеале - на 1 реальный билет).

prig\_cost2\_nodd – по каждому маршруту все деньги (выручка и потери) делятся на НОД этих сумм по всем участкам маршрута. Так как настоящий НОД вычислить получается невероятно сложно (есть такая процедура, но она просто отваливается при большом объёме данных), то вычисляется не настоящий полноценный НОД а его эрзац – в зависимости от того. на какую максимальную степень простых чисел (2,3,5,7,11,13,17) делятся все суммы по участкам. После деления остаётся уже ДОЛЯ денег, приходящихся на каждый участок маршрута.

Блок обработки видов билетов

bil\_ - перечень уникальных комбинаций параметров вида билета, имеющихся в выбранной порции данных, с указан ем на минимальное (=какое-нибудь) значение поля idnum, на котором данная комбинация была получена.

Bil – к данным BIL\_ добавляется подвыборка из bil\_ , только тех данных у которых есть признак сбора за оформление в поезде, с немного изменённым значением поля сбора в поезде.

old\_bil – выборка всех старых прочитанных вариантов видов билетов из накапливаемого справочника номеров видов билетов.

all\_bil – объединение старых данных из old\_bil, И новых данных из bil, с вычислением, имеется ли у новой записи совпадение со старой записью, чтобы взять её номер. Совершенно новым комбинациям параметра вида билета присваивается новый номер.

new\_bil\_zapis – подвыборка абсолютно новых видов билетов, для занесения в накопительный справочник.

isp\_bil – подвыборка видов билетов, имеющихся в текущей порции данных

prig\_main\_bil – склейка порции исходных данных со справочником видов билета.

Блок обработки дат отправлений.

Dats – выборка всех уникальных сочетаний типа билета, срока действия и даты начала действия.

dats\_2 – выборка, какие даты отправления в принципе попадают в период действия билетов

dats\_3 - вычисление числа поездок на каждый месяц

dats\_4 - вычисление числа поездок в каждые сутки, с учётом возможной необходимости совершения нескольких поездок в единственный день отправления последнего отчётного месяца срока действия билета.

dats\_all – совмещение результатов dats\_4 с накопительным справочником возможных разбиений дат поездок.

dats\_hash – вычисление хэш-функции разбиения дат поездок для каждой комбинации параметров вида билета и даты отправления. То есть – каждой комбинации списка запаздываний дат отправлений (относительно даты начала действия) ставится в соответствие уникальное число, полностью её характеризующее.

dats\_hash\_new – список новых хэш-значений, присваивание им новых номеров для записи в накопительный справочник.

dats\_hash\_new\_zapis – список новых комбинаций на последующую запись в справочник

dats\_itog – Объединение старых и новых комбинаций с уже придуманным номером вида комбинации дат отправления.

dats\_all2 – то же самое, но не для названий комбинаций, а их конкретных дат отправления.

dats2 – склейка результата с основными параметрами, показывающими как склеивать справочник с порцией исходных данных.

dats\_itog2 – собственно склейка каждой строки исходной порции данных со справочником номера комбинации дат отправления.

itog – сборка итогов для записи во временную таблицу.

Со значением rez=4 идёт роспись номера строки исходных данных с номером комбинации дат отправления

Со значением rez=1 идёт роспись самих уникальных новых комбинаций дат отправления

Со значением rez=2 идёт роспись уникальных новых комбинаций параметров вида билета

Со значением rez=3 идёт роспись номера строки исходных данных с номером вида билета

Со значением rez=5 идёт роспись распределения долей стоимости поездки по участкам маршрута

Со значением rez=0 идёт роспись фамилий для реестра льготников

Последний шаг – запись из выборки Itog во временную таблицу по пригороду.

1. Программа svod.prig\_obog\_1p

Программа выполняет первый этап чтения из базы данных пассажирских перевозок – в точно таком же объёме, что и программа svod.prig\_obog\_1, только из других исходников, и по большей части является точной копией этой программы. Единственно, те переменные которые есть в пригородном комплекте, но в принципе отсутствуют в пассажирском – сразу заносились с пустыми значениями по умолчанию.

1. Программа svod.prig\_obog\_21 – выполняет второй этап обработки, по маршрутам следования пассажиров. Тоже разбивается на логические блоки

prig\_cost – выборка из временной таблицы данных по участкам маршрута с долями стоимости.

prig\_cost\_hash – создание хэш-функции на каждый маршрут в текущей порции данных

prig\_cost\_opis – по значению хэш-функции присваивание каждому маршруту номера уникального маршрута в текущей порции данных

prig\_opis – расписывание только уникальных маршрутов текущего блока в виде списка всех перегонов

itog – сбор результатов для записи во временную таблицу

Со значением rez=51 идёт роспись уникальных маршрутов

Со значением rez=7 идёт роспись, какой билет описывается каким уникальным маршрутом

1. Программа svod.prig\_obog\_22 – выполняет третий этап обработки, по маршрутам следования пассажиров. Тоже разбивается на логические блоки

Dates – выборка текущей загружаемой даты, по которой далее берутся справочники.

prig\_opis – выборка из временной таблицы уникальных описаний маршрутов.

marshr\_ish – выборка из справочника пригородных маршрутов (prig.submars) действительных на текущий момент данных.

stan– выборка из справочника станций (nsi.stanv) действительных на текущий момент данных.

lin\_ish– выборка из справочника линий (nsi.lines) действительных на текущий момент данных – перечня станций с указанием их расстояний в линии.

Lin – добавление к номеру станции в линии её реквизитов из справочника станций

linp\_1 – выборка всех перегонов линий – промежутков между двумя соседними станциями, в направлении от первой станции к последней. Вычисление атрибутов принадлежности перегона (субъекта, отделения дороги, ДЦС).

Linp – то же самое, только перегоны уже в обе стороны, с указанием направления перегона в линии.

Dorlin – справочник всех линий во всех дорогах.

mar\_n – список позиций, на которых в маршруте меняется код региона

marshr – только нужные маршруты (имеющиеся в списке описаний текущей порции), с признаком смены региона по данной станции.

Sts – список уникальных возможных участков маршрута пассажира, из порции данных.

sts1\_0 as – подвыборка, список всех вариантов, найденных в справочнике маршрутов.

stan\_bad – перечень станций участков движения пассажиров, отсутствующих в справочнике маршрутов.

sts1 - если внутри маршрута (sts1\_0) несколько вариантов - оставить 1 вариант, лучший по расстоянию.

sts\_good - список вариантов (sts1), укладывающихся строго в 1 регион

sts2 - нужные к поиску пары станций (из sts1), пронумерованы.

mars\_p as --пары соседних станций внутри маршрута, с вычислением региона и МЦД

sts\_3 - нужные к поиску пары станций, в направлении маршрута

umn -роспись всех участков движения, только в направлении маршрута (склейка таблиц sts\_3 и mars\_p)

umn\_1 – к результату найденных маршрутов umn добавил все ненайденные (по причине неправильного маршрута, отсутствия в нём станции, либо чего угодно иного) – в результате маршруты движения пассажиров расписаны в терминах маршрутов электричек, с большим количеством пропущенных станций, и лишь те в которых не было ошибок.

stsm – выборка уникальных пар соседних станций маршрута пассажира, после расписывания по маршрутам поездов.

stsm\_1 – какие из пар станций (stsm) смогли найти внутри 1 линии, минимальное отличие расстояний

lin2 – перечень пересечений всех линий по какой-либо станции

stsm\_2 - поиск (из stsm), какие ещё не обнаружены по точному совпадению расстояний, с каким-нибудь проездом по 2 линиям.

stsm\_2\_1 - преобразование найденного (stsm\_2) к удобному списочному виду

stsm\_3 - поиск (из stsm), какие ещё не обнаружены по точному совпадению расстояний (по 1 или 2 линиям), с каким-нибудь проездом по 3 линиям.

stsm\_3\_1 - преобразование найденного (stsm\_3) к удобному списочному виду

stsm\_4 - поиск (из stsm), какие ещё не обнаружены по точному совпадению расстояний (по 1 или 2 или 3 линиям), с каким-нибудь проездом по 4 линиям.

stsm\_4\_1 - преобразование найденного (stsm\_4) к удобному списочному виду.

stsm\_p\_0 - объединение всех найденных (из stsm\_1, stsm\_2\_1, stsm\_3\_1, stsm\_4\_1)

stsm\_p – из объединения для каждого рассматриваемого участка выбирается наилучшее из описаний в терминах линий (с наименьшим отличием от указанного исходно расстояния по участку маршрута).

umn\_2 - полная роспись всех кусков, в терминах кусков линий

stsm\_rez – переход от кусков линий к мельчайшим перегонам в составе линий.

umn\_3 – добавка к результату перегонов, которые не удалось описать в терминах линий.

prig\_opis\_mars - роспись всех маршрутов пассажиров (а не участков маршрута) в виде минимальных перегонов.

err\_opis\_mars – список уникальных обнаруженных ошибок по принадлежности регионам.

itog – сбор результатов для записи во временную таблицу

Со значением rez=6 идёт роспись уникальных маршрутов пассажира в терминах перегонов по линиям (про номера маршрутов поездов теперь забываем)

Со значением rez=99 идёт роспись обнаруженных ошибок

1. Программа svod.prig\_obog\_3 – выполняет последний этап обработки маршрутов, вычленение уникальных и присваивание их номеров исходным строкам данных. Тоже разбивается на логические блоки

Dates – выборка читаемой даты

prig\_opis\_mars – чтение из временной таблицы описаний маршрутов в виде минимальных перегонов

prig\_cost\_opis as -- чтение из временной таблицы соответствий (строка данных - номер описания)

prig\_opis\_hash – создание хэш-функций полных росписей маршрутов для каждого описания маршрута из текущей порции данных

mars\_opis\_hash – создание точно таких-же хэш-функций полных росписей маршрутов, из уже накопленного справочника маршрутов за прошлые периоды.

prig\_opis\_nom\_mar – выяснение, какие из новых маршрутов прежде уже встречались, и им взять их старый номер, а какие совершенно новые – им присвоить новый номер.

prig\_opis\_nom\_mar\_new – выборка номеров новых маршрутов

new\_mar – выборка описаний новых маршрутов, для последующего занесения в накопительный справочник.

prig\_cost\_opis\_mar – присваивание строкам данных номеров маршрутов

prig\_mar\_bil\_dat – подтягивание из временной таблицы к номеру строки ещё и номера вида билета, номера распределения дат поездок, и всех стоимостных показателей.

itog – сбор результатов для записи во временную таблицу

Со значением rez=9 идёт роспись итоговых данных – какой номер строки какие имеет парараметры по маршруту, виду билета и датам отправления.

Со значением rez=8 идёт роспись новых маршрутов для занесения в справочник.

1. Программа svod.prig\_obog\_4 – выполняет запись рассчитанных данных во все справочники и агрегированные данные. Она разбивается на программные блоки.

Блок 1 – запись в справочник распределения дат отправления из временной таблицы, берутся строки с признаком rez=1.

Блок 2 – запись в справочник новых видов билетов из временной таблицы, берутся строки с признаком rez=2.

Блок 3 – запись в справочник номеров перегонов, из временной таблицы берутся строки с признаком rez=8 (=маршруты), из них вычленяются все перегоны (промежутки между двумя подряд идущими станциями), и те которых раньше не было дописываются в справочник l3\_prig.prig\_peregoni. Данная процедура нужна для последующих справок о пассажиропотоке по конкретным перегонам и более крупным участкам компаний.

Блок 4 – запись в справочник новых видов маршрутов из временной таблицы, берутся строки с признаком rez=8, и на лету приклеиваются номера перегонов.

Блок 5 – запись в справочник обнаруженных ошибок маршрутов из временной таблицы, берутся строки с признаком rez=99.

Блок 6 – запись в таблицу агрегированных данных из временной таблицы, берутся строки с признаком rez=9. Именно здесь все наработанные данные наконец проходят собственно агрегацию (суммирование) по уникальным комбинациям вида билета, даты продажи и оформления, распределения дат отправления, маршруту, атрибутам компании и станции продажи билета, а так же по цене билета (два одинаковых билета, но с разной стоимостью проезда – не просуммируются в одну строку).

Блок 7 – запись в таблицу реестра льготников, из временной таблицы берутся строки с признаком rez=9, все параметры включая ФИО, и по справочнику видов билетов выбираются лишь те, которые нужны в реестре по совокупности признаков льгот. После этого происходит переработка данных до требуемой в реестре структуры записи, с использованием справочника дорог, станций, перевода кодов субъектов федерации в коды ОКАТО, и нескольких дополнительных прочих.

Блок 8 – удаление из таблицы статистики по реестру льготников итогов за отчётные месяца, которые были в перечне дозагруженных данных.

Блок 9 – несколько шагов по поиску и пометке на удаление пар записей продажа-возврат в реестре льготников в загруженных данных.

Блок 10 – заполнение таблицы статистики по реестру льготников итогов за отчётные месяца, которые были в перечне дозагруженных данных.

1. Программа svod.prig\_agr\_analit – наработка аналитической отчётности из новых наработанных порций агрегированных данных по пригороду (имеется в виду только пригородные данные, без пассажирских по 8\*\*-АГМХИЙ поездам).
2. Программа svod.prig\_co22 – наработка аналитической отчётности из новых наработанных порций агрегированных данных по пригороду, включая 8\*\*-АГМХИЙ поезда.

Программа формирует данные в несколько программных блоков, с промежуточной записью данных в таблицу временных данных по ЦО-22 (l3\_mes.prig\_co22\_t0)

Основной блок – расчёт 6 таблиц данных для макета ЦО-22 (таблицы 1,2,3,4,6, без таблицы 5), с занесением итогов во временную таблицу.

Вспомогательные блоки – занесение в постоянные таблицы (l3\_mes.prig\_co22\_t1, l3\_mes.prig\_co22\_t2, l3\_mes.prig\_co22\_t3, l3\_mes.prig\_co22\_t4, l3\_mes.prig\_co22\_t6) рассчитанных итогов по обработанным данным. В таблицу l3\_mes.prig\_co22\_t5 результирующая статистика заносится по результатам вычисления остальных таблиц.

Основной блок программы вычислений делится на логические блоки:

Prig – чтение данных из таблицы пригородных агрегатов за конкретный отчётный месяц.

Dates – выбор максимальной даты загрузки, в качестве актуальной даты для справочников.

Stan – выборка из справочника станций по актуальной дате.

dat1 – уникальные сочетания даты продажи, начала действия и номера распределения дат отправления.

dat2 – подтягивание справочника дат отправления

dat3 – приведение получившегося результата к виду удобному для склейки с исходными данными. По дате начала действия и запаздыванию вычисляется дата отправления и количество поездок и проданных билетов в каждый день отправления. Вычисление года и месяца отправления пассажира.

Dor – справочник дорог по актуальной дате.

Dor2 – в справочник дорог по номеру ВЦ добавляется код дороги ВЦ.  
prig1 – соединение исходных данных со справочником дорог по коду терминала продажи билета.

OKATO – справочник кодов ОКАТО – актуальный вариант, то есть действующий в бесконечности.

mar1 – список всех маршрутов из справочника, имеющихся в выбранной порции данных.

mar2 – вычисление по маршрутам станций начала и окончания, полного расстояния, а так же дороги, отделения и региона станций начала и конца маршрута, максимального номера МЦД в маршруте, и полного расстояния проследования по непустому МЦД.

Mar – к результату mar2 приклеиваются по станции начала и конца маршрута коды ОКАТО и nopr

Bil – блок выборки из справочника видов билетов всех нужных для ЦО-22 параметров, в требуемом виде.

itog1 – формирование всех данных для главной таблицы отчёта ЦО-22 в сыром виде (не ужатом)

itog2\_0 – итоги просуммировали по уникальным комбинациям показателей, и ввели номера строк – нумерация несплошная, с нарушением уникальности для строк данных по одинаковым билетам с разными датами отправления.

itog2 – Нумерация строк делается сквозной, без пробелов.

mar\_otd – внутри маршрута создаётся нумерация по сменяющимся отделениям

itog\_mar\_otd – к таблице итогов приклеиваются маршруты по отделениям (вторая таблица макета ЦО-22).

mar\_reg – аналогично в маршрутах создаётся нумерация по регионам

itog\_mar\_reg - к таблице итогов приклеиваются маршруты по регионам (третья таблица макета ЦО-22).

mar\_dcs – аналогично в маршрутах создаётся нумерация по ДЦС

itog\_mar\_dcs - к таблице итогов приклеиваются маршруты по ДЦС (шестая таблица макета ЦО-22).

mar\_nomera – обработка маршрутов для вычисления разбивки денег и потерь по маршруту следования.

mar\_pl – присоединение к маршрутам справочника ОКАТО, и дальнейшая обработка маршрутов.

itog\_plat\_ - разбивка денег по каждому билету, точная, но вычислительно очень длительная.

itog2\_plat\_ - аналогичное разбиение денег из уже агрегированных данных, посему более приближённое, но гораздо быстрее считается.

itog\_plat – итоговое вычисление данных для 4 таблицы макета ЦО-22 по разбиению денег (из приближённой, но быстрой процедуры вычисления).

itog\_5 – подведение статистики таблиц 1-4, для таблицы 5.

Rezult – сборка итогов для записи во временную таблицу.

В конце – запись во временную таблицу всех собранных результатов.

Часть 2. Своды

Файл prog\_log\_svod (программа PROCEDURE svod.svod\_dann\_poln(load\_ text) ) – запускает загрузку всех сводов по пассажирским перевозкам. Последовательный запуск в цикле программ:

call svod.svod\_bag(load\_); - чтение порции данных по багажу

call svod.svod\_pas(load\_); - чтение порции данных по пассажирам

call svod.svod\_krs(load\_); - чтение порции данных по КРС

call svod.svod\_meal(load\_); - чтение порции данных по питанию

call svod.svod\_card(load\_);- чтение порции данных по картам

Эти программы сформированы в файлах prog\_log\_svod\_bag, prog\_log\_svod\_pas, prog\_log\_svod\_krs, prog\_log\_svod\_meal, prog\_log\_svod\_card.

1. Программа svod.svod\_bag – чтение порции данных для сводов по багажу.

Программа за указанную дату делает полную обработку данных со 2 уровня до состояния загрузки в реестр багажа, с использованием временной таблицы l3\_mes.svod\_work. Состоит из логических блоков

Dats – выбор нужной даты чтения.

Gos, Dor – справочники государств и дорог актуальный на выбранную дату.

Main – исходный объём данных из таблицы rawdl2m.l2\_bag\_main

Cost – исходный объём дополнительных данных распределения стоимостей из таблицы rawdl2m.l2\_bag\_cost

Dann – обработка данных выборки main с использованием всех справочников.

grup1, grup2 – в 2 этапа обработка dann с целью сгруппировать данные и создать нужную нумерацию строк, начиная с последнего номера предыдущей порции чтения (номер прочитан из справочника).

grup3 – выяснение соответствия нового номера сгруппированных строк старым номерам строк исходных данных.

cost2 – к распределению стоимостей применяется такая же группировка, какая получилась с исходными данными.

Itog – сборка результатов с разными значениями поля rez

При rez=1 записываются сгруппированные данные – далее они идут в таблицу rawdl2\_day.svod\_bag\_main

При rez=2 записываются соответствия исходных и сгруппированных номеров строк – далее они идут в таблицу rawdl2\_day.link\_svod\_bag\_main

При rez=3 записываются переработанные распределения стоимостей. – далее они идут в таблицу rawdl2\_day.svod\_bag\_cost

1. Программа svod.svod\_pas – чтение порции данных для сводов по скоростных электричек с местами.

Программа за указанную дату делает полную обработку данных со 2 уровня до состояния загрузки в реестр пассажиров, с использованием временной таблицы l3\_mes.svod\_work. Состоит из логических блоков

Dats – выбор нужной даты чтения.

Gos, Dor – справочники государств и дорог актуальный на выбранную дату.

Pass – исходный объём данных из таблицы rawdl2m.l2\_pass\_main

Cost1 – исходный объём дополнительных данных распределения стоимостей из таблицы rawdl2m.l2\_pass\_cost

Bilgr – дополнительные данные по льготам и билетным группам из таблицы rawdl2m.l2\_pass\_ex

Dann – обработка данных выборки pass с использованием всех справочников.

grup1, grup2 – в 2 этапа обработка dann с целью сгруппировать данные и создать нужную нумерацию строк, начиная с последнего номера предыдущей порции чтения (номер прочитан из справочника).

grup3 – выяснение соответствия нового номера сгруппированных строк старым номерам строк исходных данных.

cost2 – к распределению стоимостей применяется такая же группировка, какая получилась с исходными данными.

Itog – сборка результатов с разными значениями поля rez

При rez=1 записываются сгруппированные данные – далее они идут в таблицу rawdl2\_day.svod\_pass\_main

При rez=2 записываются соответствия исходных и сгруппированных номеров строк – далее они идут в таблицу rawdl2\_day.link\_svod\_pass\_main

При rez=3 записываются переработанные распределения стоимостей. – далее они идут в таблицу rawdl2\_day.svod\_pass\_cost

1. Программа svod.svod\_krs – чтение порции данных для сводов по КРС.

Программа за указанную дату делает полную обработку данных со 2 уровня до состояния загрузки в реестр КРС, с использованием временной таблицы l3\_mes.svod\_work. Состоит из логических блоков

Dats – выбор нужной даты чтения.

Gos, Dor – справочники государств и дорог актуальный на выбранную дату.

Krs – исходный объём данных из таблицы rawdl2m.l2\_krs

Dann – обработка данных выборки krs с использованием всех справочников.

grup1, grup2 – в 2 этапа обработка dann с целью сгруппировать данные и создать нужную нумерацию строк, начиная с последнего номера предыдущей порции чтения (номер прочитан из справочника).

grup3 – выяснение соответствия нового номера сгруппированных строк старым номерам строк исходных данных.

Itog – сборка результатов с разными значениями поля rez

При rez=1 записываются сгруппированные данные – далее они идут в таблицу rawdl2\_day.svod\_krs

При rez=2 записываются соответствия исходных и сгруппированных номеров строк – далее они идут в таблицу rawdl2\_day.link\_svod\_krs

4.5. Программы call svod.svod\_meal- чтение порции данных по питанию, и call svod.svod\_card – чтение порции данных по картам.

Обе программы совершенно идентичны программе svod.svod\_krs, писались как её копии, только с выборкой из других таблиц, но с той же самой логической структурой, и с записью результатов в подобные таблицы. Разница только в структуре выдаваемых данных.