Техническое задание на разработкухранилища данных. Этап 1.

Введение.

Сегодня ваш первый день работы инженером по данным в компании такси “Везу и точка”. К вам приходит начальник и говорит тоном, не терпящим возражений: “Что это мы живём без нормальных отчётов? Мне нужно, чтобы ты закончил работу по их созданию”. После чего подмигивает и уходит. От коллеги справа, который работает с операционной системой, в которой ведётся деятельность компании, вы узнаёте, что ваш предшественник начал делать эту задачу, но внезапно решил уволиться.

Открыв папку с документацией, где предшественник всё хранил, вы находите документ, который описывает источник, актуальность которого подтверждает всё тот же коллега справа, а также описание целевой структуры хранилища. Чтобы не терять времени вы решаете использовать её.

Ваша первая задача - настроить ETL процесс, который будет забирать данные из источника и раскладывать их по целевым таблицам в описанную в документации структуру в хранилище.

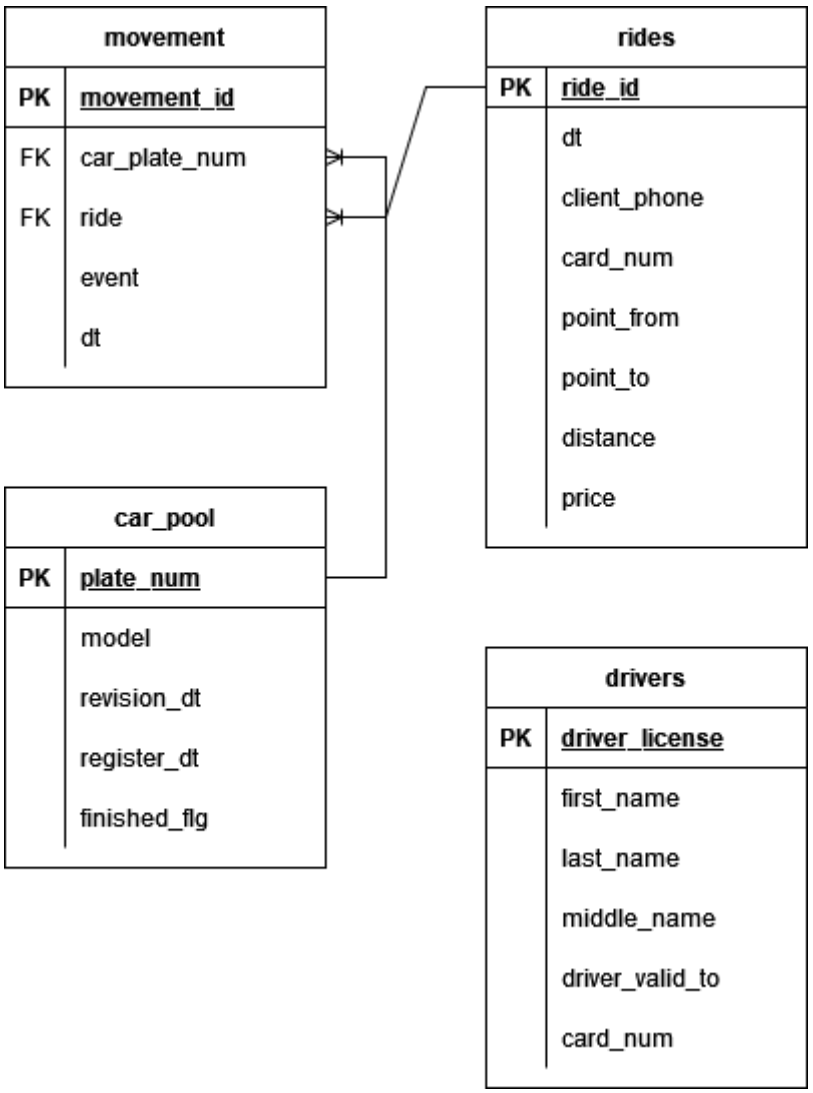
Описание источников данных.

1. Источник данных - СУБД PostgreSQL

Реквизиты подключения к источнику техническим пользователем:

- host: de-edu-db.chronosavant.ru
- port: 5432
- database: taxi
- schema: main
- user: etl_tech_user
- password: etl_tech_user_password

ER-диаграмма.



Описание сущностей.

Таблица main.rides

Заказы, поступающие от клиентов.

PK	NAME	TYPE	DESCRIPTION
PK	ride_id	INTEGER	Идентификатор заказа-поездки. Автоинкремент.
	dt	TIMESTAMP(0)	Дата и время заказа.
	client_phone	CHAR(18)	Номер телефона клиента.
	card_num	CHAR(19)	Карта клиента *(обратите внимание, при разных заказах одного и того же клиента может быть разной).*

	point_from	VARCHAR(200)	Начальный адрес.
	point_to	VARCHAR(200)	Конечный адрес.
	distance	NUMERIC(5,2)	Дистанция поездки.
	price	NUMERIC(7,2)	Цена поездки.

Таблица main.movement

Статусы машин, направленных на заказ. Каждый статус добавляется новой строкой с временно́й меткой. При успешно выполненном заказе по одному заказу будет три записи - подача машины (READY), начало поездки (BEGIN) и окончание поездки (END). Отмена заказа (CANCEL) возможна только до его начала, при этом завершения заказа (END) не будет.

PK	NAME	TYPE	DESCRIPTION
PK	movement_id	INTEGER	Идентификатор факта перемещения. Автоинкремент.
FK	car_plate_num	CHAR(9)	Номер машины.
FK	ride	INTEGER	Ссылка на заказ-поездку.
	event	VARCHAR(6)	Произошедшее событие: READY - машина подана BEGIN - начало выполнения заказа CANCEL - отмена заказа (при таком статусе ENDне будет) END - завершение заказа (если он начал выполняться)
	dt	TIMESTAMP(0)	Дата и время события.

Таблица main.car_pool

Машины, зарегистрированные в автопарке. Машина регистрируется при первом участии ее в любой поездке. Каждые три дня машина должна проходить осмотр, отметка о проведении осмотра *обновляется* в таблице. Если машина больше не работает с компанией - обновляется соответствующий флаг (в данном кейсе таких ситуаций нет).

PK	NAME	TYPE	DESCRIPTION
PK	plate_num	CHAR(9)	Номер автомобиля
	model	VARCHAR(30)	Марка автомобиля
	revision_dt	DATE	Дата последнего осмотра
	register_dt	TIMESTAMP(0)	Дата первой регистрации машины в парке

	finished_flg	CHAR(1)	Машина списана (больше не работает в парке):Y/N

Таблица main.drivers

Водители, работающие в компании. Водители могут добавляться независимо от поездок. Иногда у водителя может меняться номер карты, это изменение фиксируется обновлением соответствующего поля.

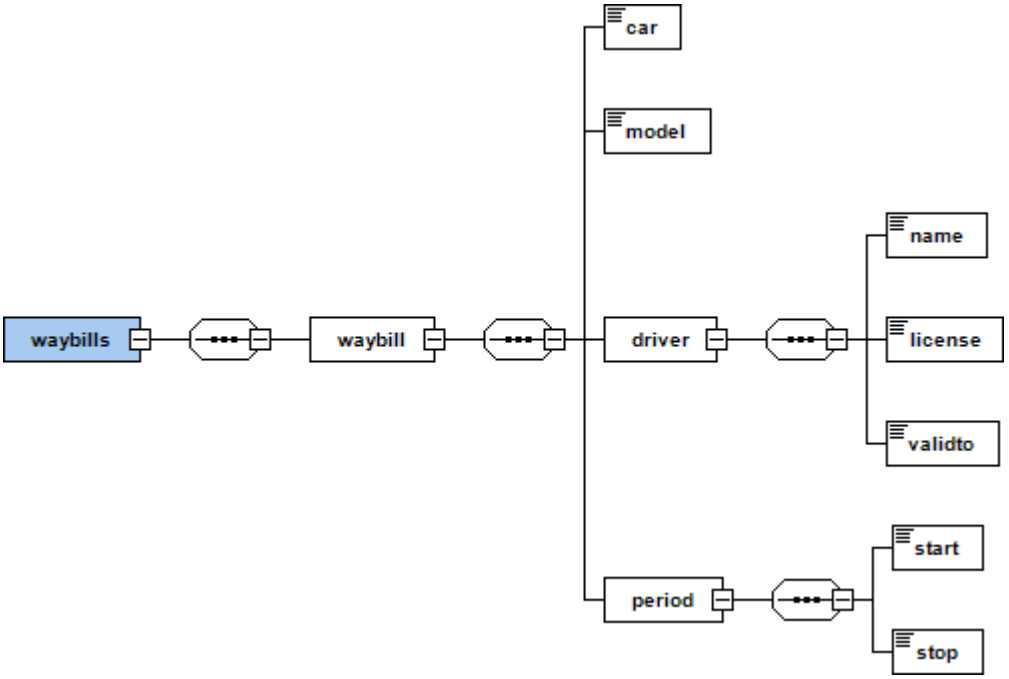
PK	NAME	TYPE	DESCRIPTION
PK	driver_license	CHAR(12)	Номер водительского удостоверения
	first_name	VARCHAR(20)	Фамилия
	last_name	VARCHAR(20)	Имя
	middle_name	VARCHAR(20)	Отчество
	driver_valid_to	DATE	Срок действия водительского удостоверения
	card_num	CHAR(19)	Номер карты водителя
	update_dt	TIMESTAMP(0)	Дата и время обновления записи

2. Источник данных - файловые выгрузки.

Каталог файловых выгрузок примонтирован к /mnt/files

Каталог waybills содержит путевые листы, в одном файле один путевой лист. Путевые листы содержат информацию о том, какой водитель в какой период времени на какой машине работал.
Файлы выгружаются по факту появления. Маска имени waybill_NNNNNN.xml, номер NNNNNN возрастает по порядку начиная с 1.

Структура файла:



Каталог payments содержит бухгалтерские выписки о поступлении денег от клиентов на счет компании. Файлы выгружаются раз в полчаса. Маска имени payment_YYYY-MM-DD_HH-MI.csv.

Структура файла (разделители - табуляция):

Поле	**Формат**	**Длина**
Дата и время	DD.MM.YYYY HH24:MI:SS	19
Номер карты	NNNNNNNNNNNNNNN	16
Сумма платежа	##N.NN	4-6

Требования к хранилищу данных.

На первом этапе от вас требуется создать детальный слой хранилища данных. В детальном слое обычно создаются таблицы, описывающие каждую бизнес-сущность, существующую в компании. В нашем случае это будут: заказы (поездки), клиенты, машины, водители и платежи. Также необходима техническая таблица, связывающая водителей и машины - в нашем случае это информация о путевых листах.

Приемник данных (хранилище) - СУБД PostgreSQL

Реквизиты подключения к хранилищу техническим пользователем:

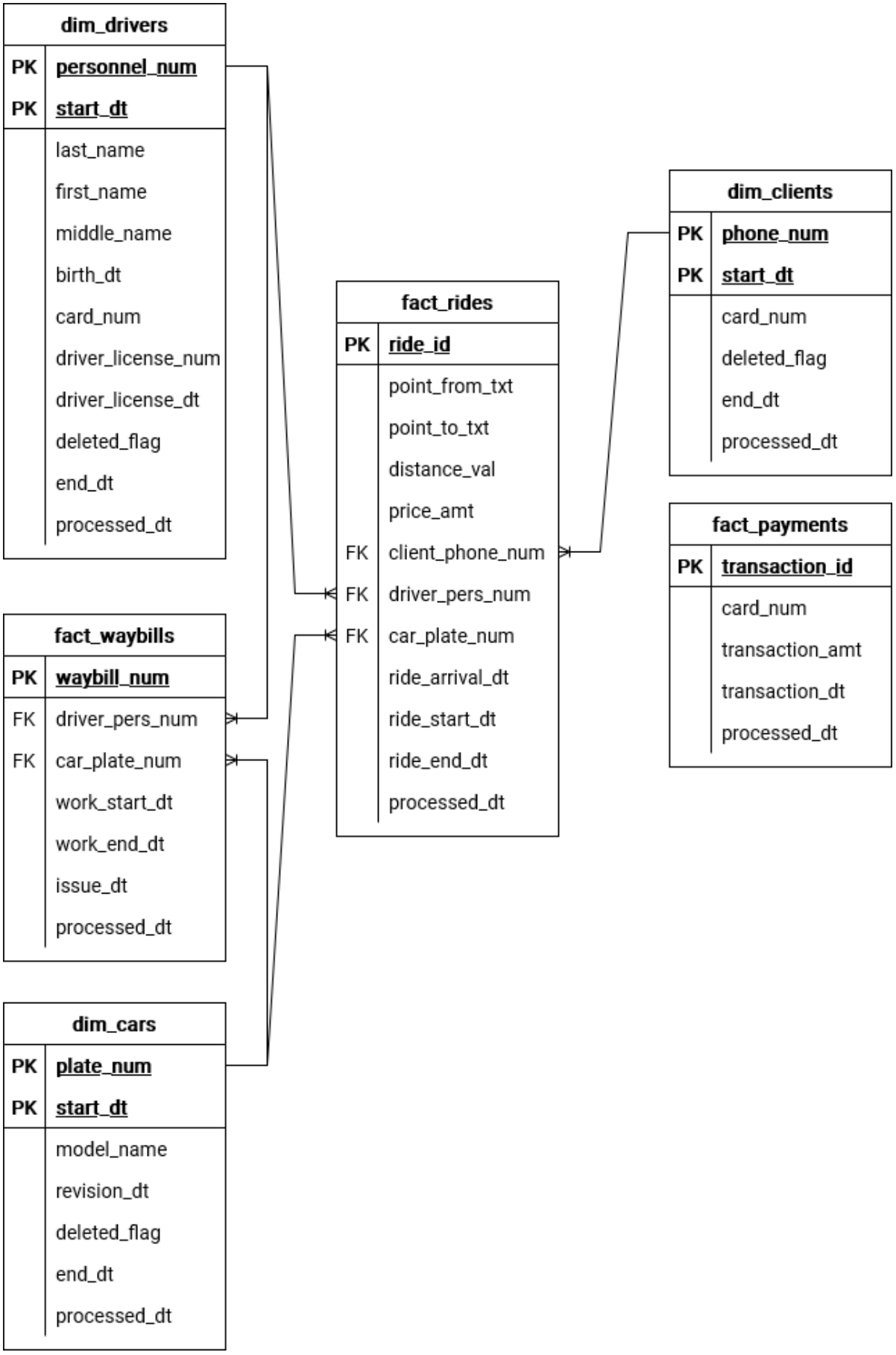
- host: de-edu-db.chronosavant.ru
- port: 5432
- database: dwh
- schema: dwh_<имя_вашего_города>
- user: dwh_<имя_вашего_города>
- password: <ваш_пароль>

*Наименование схемы, пользователя и пароль выдает ментор

ER-диаграмма целевого хранилища данных.

Обратите внимание что вам необходимо строго следовать правилам именования всех объектов. Типизацию выбирайте из соображений разумной потребности (например, поле last_name вряд ли стоит назначать типом DATE, разумнее выбрать VARCHAR. Длину поля тоже имеет смысл сделать 50 символов, а не 25000, это, кстати, может быть поводом для отдельного исследования).

В хранилищах данных, как правило, не назначаются ограничения целостности типа unique, pk, fk (из-за их большой ресурсоемкости), поэтому проверка constraints непредусмотрена.


dim_drivers
PK personnel_num
PK start_dt
last_name
first_name
middle_name
birth_dt
card_num
driver_license_num
driver_license_dt
deleted_flag
end_dt
processed_dt
fact_rides
PK ride_id
point_from_txt
point_to_txt
distance_val
price_amt
FK client_phone_num
FK driver_pers_num
FK car_plate_num
ride_arrival_dt
ride_start_dt
ride_end_dt
processed_dt
dim_clients
PK phone_num
PK start_dt
card_num
deleted_flag
end_dt
processed_dt
fact_payments
PK transaction_id
card_num
transaction_amt
transaction_dt
processed_dt
fact_waybills
PK waybill_num
FK driver_pers_num
FK car_plate_num
work_start_dt
work_end_dt
issue_dt
processed_dt
dim_cars
PK plate_num
PK start_dt
model_name
revision_dt
deleted_flag
end_dt
processed_dt

Фактовые таблицы:

- fact_rides
- fact_payments
- fact_waybills

Таблицы-измерения:

- dim_clients
- dim_cars
- dim_drivers

Заполнение фактовой таблицы fact_rides должно производиться только для завершенных поездок. То есть в источнике для поездки должен стоять либо статус END, либо статус CANCEL. Незавершенные поездки недопустимы. Правила заполнения полей следующие:

- ride_arrival_dt - время прибытия машины на заказ (статус READY),
- ride_start_dt - время начала заказа (статус BEGIN, если заказ был отменен - то NULL),
- ride_end_dt - время завершения или отмены заказа (статус END или CANCEL).

Во всех таблицах измерений должны быть корректно заполнены версии, то есть

- поля, определяющие ключ
- start_dt
- end_dt.

Техническое задание на разработкухранилища данных. Этап 2.

Введение.

Итак, вы завершили первый этап. Задача второго этапа - получить бизнес-результат по данным, загруженным в хранилище. Для этого требуется составить четыре отчета, каждый из которых сложнее предыдущего.

Требования к отчетам.

Для ежедневных витрин необходимо построить полный отчет за завершенный день. При этом в одной таблице хранятся отчеты за все дни, у каждого из них свое значения даты, на которую построен отчет - report_dt (обратите внимание, это не то же самое что дата построения отчета).

Если вы считаете, что требования неточные - требуется задать уточняющие вопросы заказчикам (менторам).

1. Выплата водителям (rep_drivers_payments).

- Ежедневная витрина.
- Критерии построения:
    - услуги сервиса - 20% от стоимости поездки,
    - стоимость топливо - 47,26 * 7 * дистанция / 100,
    - амортизация машины 5 руб / км * дистанция,
    - остаток достается водителю.
- Атрибуты таблицы:
    - personnel_num - табельный номер водителя,
    - last_name - фамилия,
    - first_name - имя,
    - middle_name - отчество,
    - card_num - номер карты водителя,
    - amount - сумма к выплате,
    - report_dt - дата, на которую построен отчет.

2. Водители-нарушители (rep_drivers_violations).

- Ежедневная витрина.
- Критерии:
  - завершил поездку со средней скоростью более 85 км/ч.
- Атрибуты:
  - personnel_num - табельный номер водителя,
  - ride - идентификатор поездки,
  - speed - средняя скорость,
  - violations_cnt - количество таких нарушений, выявленных ранее, накопленной суммой. Первое нарушение - значение 1
  - report_dt - дата, на которую построен отчет.

3. Перерабатывающие водители (rep_drivers_overtime).

- Ежедневная витрина.
- Критерии:
  - водитель работал больше 7 часов за 24 часа (по путевым листам).
  - нарушением считается тот путевой лист, который вышел за пределы 8часов. Таких путевых листов может быть много.
  - дата нарушения - по дате начала работы в путевом листе.
- Атрибуты:
  - personnel_num - табельный номер водителя,
  - дата-время старта 24-часового периода, где он работал больше 8 часов
  - суммарная наработка
  - report_dt - дата, на которую построен отчет.