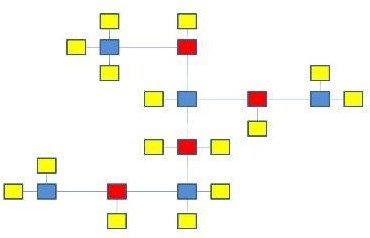
Софийски университет „Свети Климент Охридски“ Факултет по математика и информатика



# Документация към проект по дисциплина

**„Складове от данни и бизнес анализ -**

**основи“**

DATA WAREHOUSE OF CORE BANKING SYSTEM (CBS)

Изготвено от:

Екатерина Коджабашева 9MI0600121

Калин Костадинов 1MI0600085

Антония Христова 5MI0600065

Марио Симеонов 7MI0700043

Проверяващ: Тодор Кичуков

# Съдържание

Въведение ……………………………………………………..……. 3

Банковата система………………………………………………….. 4

Source data Model……………………………………………….4

Описание на система………………………………………………...5

Подходът, който избрахме………………………………………….6

Staging area Model…………………………………………………….7

ЕTL ……………………………………………………………………..8

Data warehouse approach justification……………………………. 9

Data warehouse dimensional model…………………………..........9

Data Marts……………………………………………………………10

Queries…………………………………………………………...……14

# Въведение

Следният проект се осъществява, поради причината, че банка възложи задача на нашия екип да анализираме данните на тяхната банкова система и да изградим склад от данни, който да е подходящ за извършване на анализи.

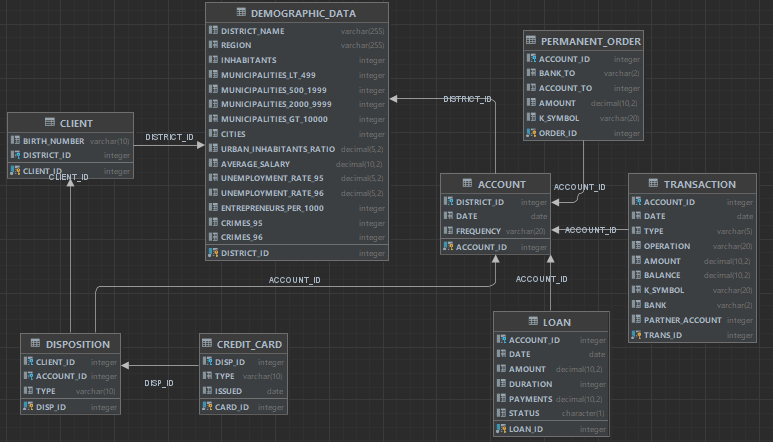
Банката ни предоставя модел на банковата система, както и описанието й.

В документацията ще се включват: описание на задачата, описание на банковата система, графично визуализиране на данните през различините етапи на разработката (Staging Area data model, Source data model, Data Warehouse data model), както и описание на подходът, по който сме избрали да работим и причините защо сме избрали него.

# Банковата система

(The Core Banking System)

Source data model



Описание на системата

Банковата система, която е обект на нашата реализация се състои от 8 основни релации.

1. „Акаунт“ с атрибути: идентификатор на акаунта(PK), идентификатор на областта, дата, честота.
2. „Клиент“ с атрибути: идентификатор на клиента(PK), дата на раждане, идентификатор на областта.
3. „Разположение“ с атрибути: идентификатор на разположението(РК), идентификатор на клиента, идентификатор на акаунта, тип на разположението.
4. „Постоянна поръчка“ с атрибути: идентификатор нa поръчката(РК), идентификатор на акаунта, банка на получател, акаунт на получател, количество, символ характеризиращ плащането.
5. „Транзакция“ с атрибути: идентификатор на транзакцията(РК), идентификатор на акаунта, дата, тип, операция, количество, баланс, символ характеризиращ транзакцията, банка, акаунт.
6. „Заем“ с атрибути: идентификатор на заема(РК), идентификатор на акаунта, дата, количество, продължителност, вноски, статус.
7. „Кредитна карта“ с атрибути: идентификатор на картата(РК), идентификатор на разположението, тип, дата на издаване.
8. „Демографски данни“ с атрибути: идентификатор на областта(РК), име на областта, регион, брой жители, брой общини с по-малко от 499 жители, брой общини с жители между 500 и 1999, брой общини с жители между 2000-9999, брой общини с повече от 10000 жители, брой градове, съотношение на градски жители, средна заплата, процент безработица през 95-та, процент безработица през 96-та, брой предприемачи на 1000 жителя, брой извършени престъпления през 95-та, брой престъпления през 96-та.

# Подходът, който избрахме

Подходът разработен от Ралф Кимбъл се фокусира на изграждането на дата уейрхауси, които са лесни за разбиране и използване от страна на бизнес потребителите.

Подходът на Кимбъл се фокусира на нуждите на бизнеса и прави данните достъпни и разбираеми за бизнес потребителите. Моделите са проектирани така, че да отразяват бизнес процесите и да улеснят анализа на данните.

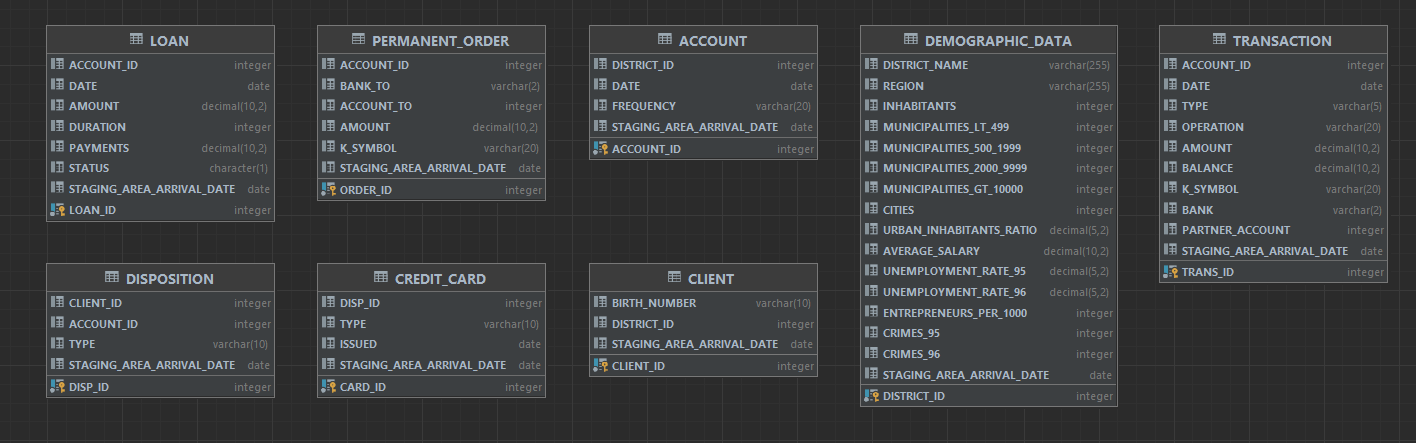
Подходът на Кимбъл обикновено включва денормализация на данните, което улеснява и ускорява изпълнението на заявки за анализ.

Моделите на Кимбъл са проектирани да бъдат устойчиви към промени в бизнес изискванията. Добавянето на нови данни или промени в бизнес правилата трябва да бъде по-лесно и по-евтино.

Подходът на Кимбъл насърчава итеративния подход при изграждането на дата уейрхауса. Това позволява на организацията да доставя бързи и конкретни резултати, докато продължава да подобрява и разширява своя дата уейрхаус.

Кимбъл поддържа използването на звездовидна схема (Star Schema), където данните са структурирани в "факти" и "измерватели". Този подход прави модела на данните лесен за разбиране и използване. Звездовидната схема и денормализацията позволяват бърз достъп до данни и ефективно изпълнение на аналитични заявки.

Staging area model



Целта ни в разработването на Staging area-та бе да създадем буфер, в който да се изсипва информацията от източниците(в нашия случай Source data, която е представена с DB Source) и временно да се съхраняват данните докато warehouse-a бъде зареден. Тя трябва да е максимално близка до Source-а. Данните първо трябва да бъдат обработени според поставените бизнес правила и форматирани по модела на Data warehouse-a. Като за пример ще бъдат добавени поле за времето на пристигане на данните.

**ETL описание:**

Извличане (Extract):

Данните се извличат от източника, за да се попълни staging областта.

Извличане на релевантна информация за всяка таблица за измерение и факт, включително данни за клиенти, сметки, заеми, транзакции, разпределения, кредитни карти и статични демографски данни за районите.

Трансформация (Transform):

Внедряваме логиката за трансформация, за да се почистят, стандартизират и засилят извлечените данни преди да се заредят в data warehouse.

Пример: обработват се липсващи стойности, конвертират се типове данни, прилагат се бизнес правила.

Създаваме заместващи ключове за таблиците за измерение (например, date\_id, time\_id, disposition\_id, credit\_card\_id и др.), за да установим връзки с таблиците за факти.

Зареждане (Load):

Зареждаме трансформираните данни в съответните таблиците за измерение и факт в data warehouse.

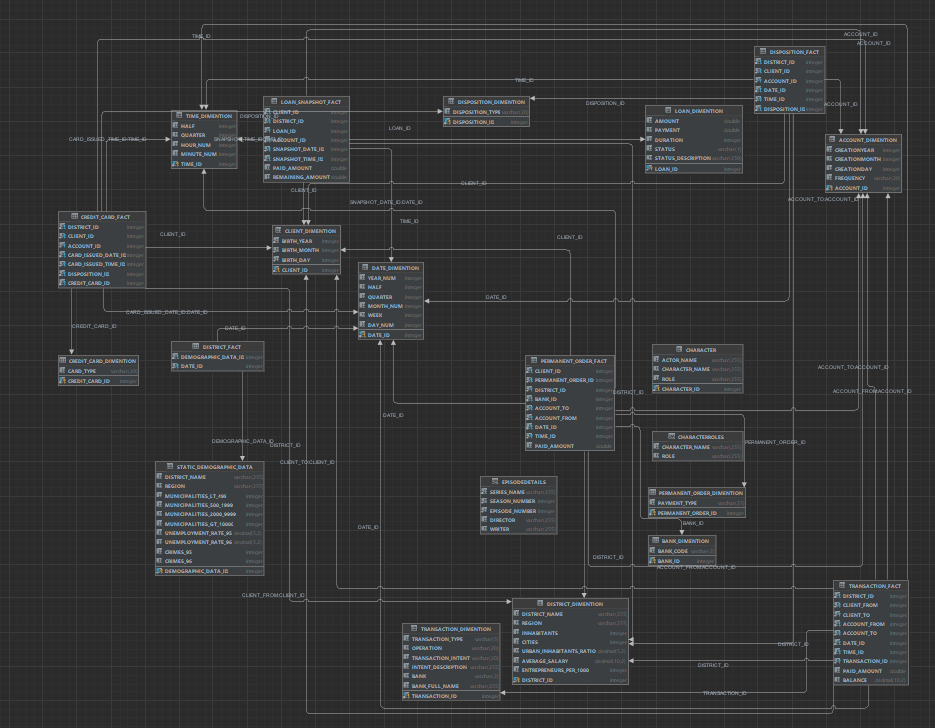
Зареждаме данните инкрементално, за да актуализираме data warehouse-a с нова информация без да зареждаме целия набор от данни отново.

ETL процесите са автоматизирани, което позволява да се изпълняват на подходящи интервали и да се поддържа актуалността на данните.

**Data warehouse Kimball model**

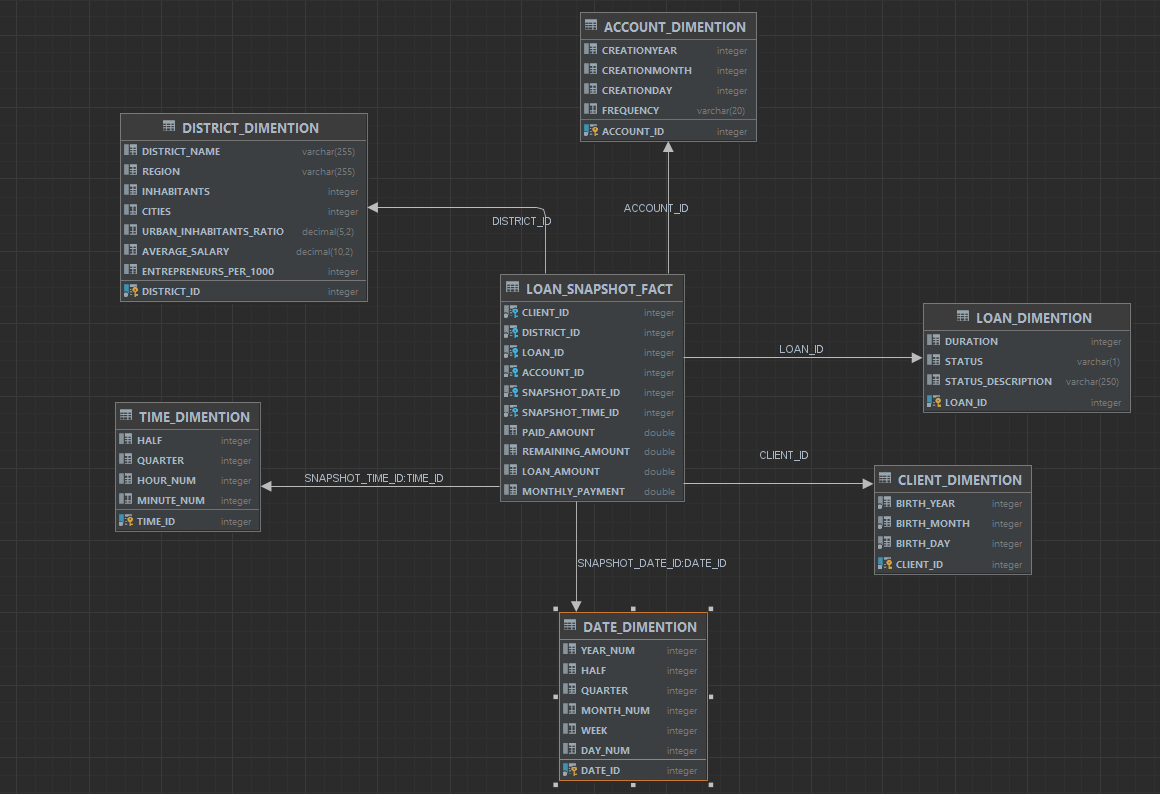
Това е графичен модел на Data Warehouse-a, което представлява нашето хранилище за данни. По пътя си от Staging\_area-та до Data Warehouse-a данните биват трансформирани и разпределени в таблици наречени “измерватели” като биват свързани във ‘’факт’’ таблици чрез техните уникални ключове, намалявайки броя различни ключове спестявайки памет и улеснявайки достъпа до информацията. При Кимбъл Data Warehouse-a е съвкупност от Data Mart-овете.

Представени са 2 различни диаграми на Data Warehouse-a:

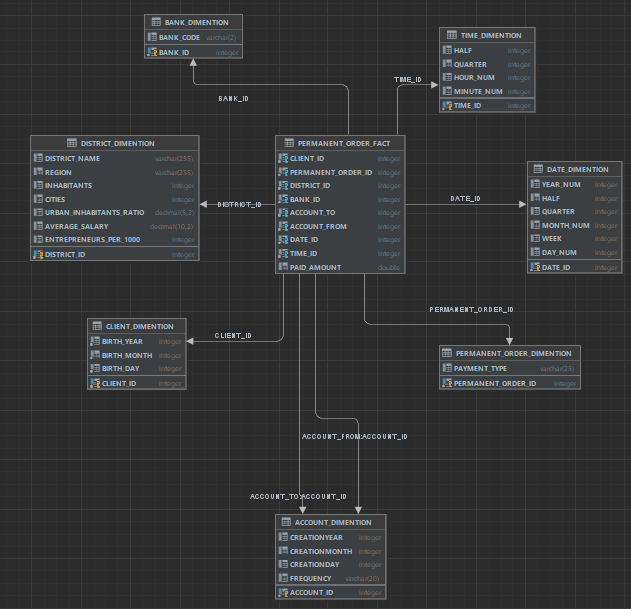


Data Marts

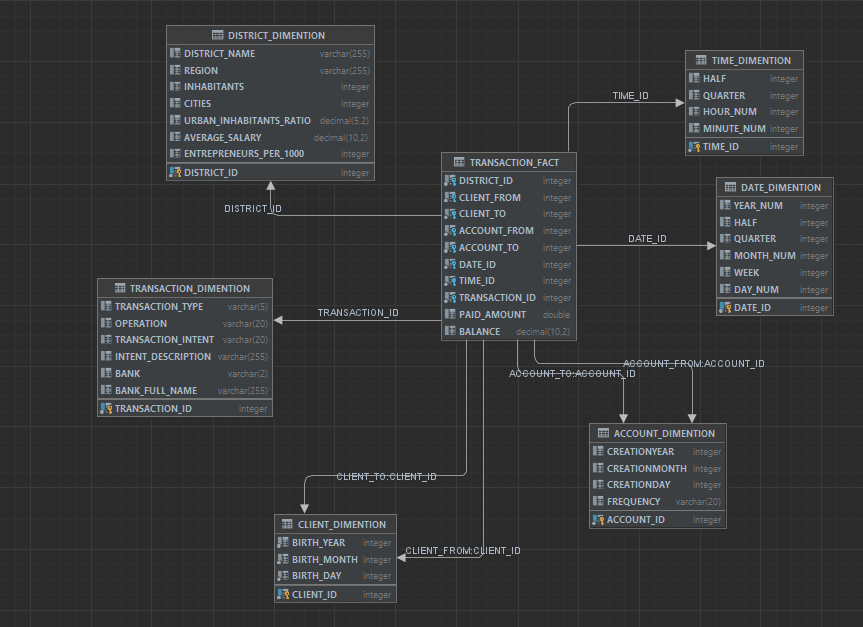
Grain: дневен снапшот

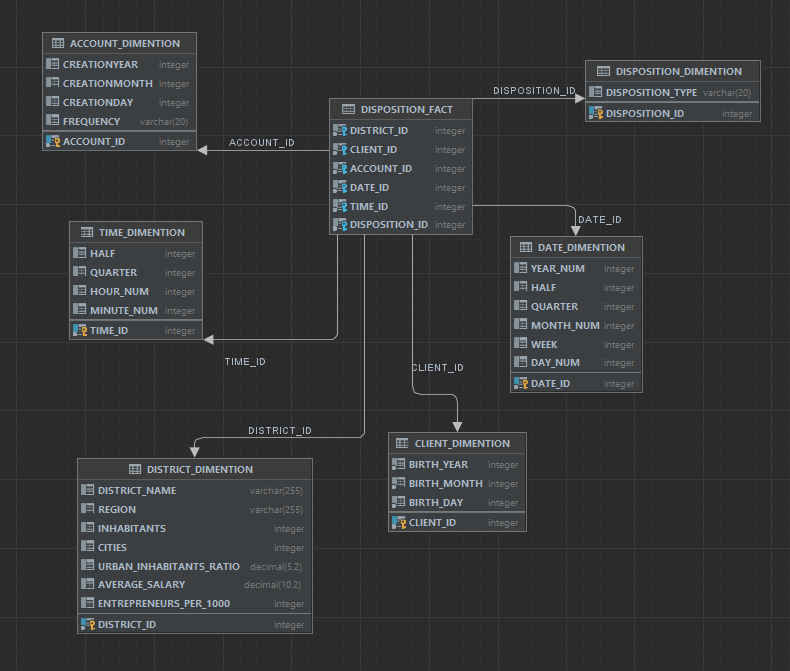


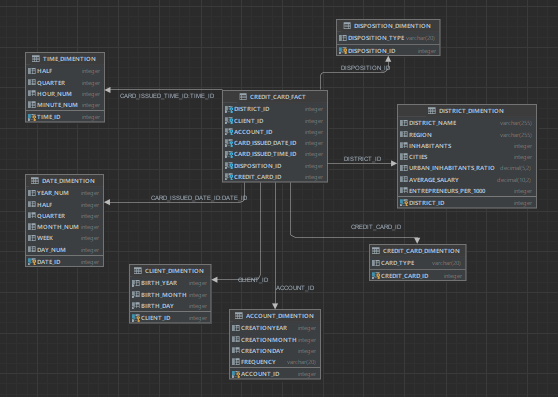
Grain: Една Поръчка

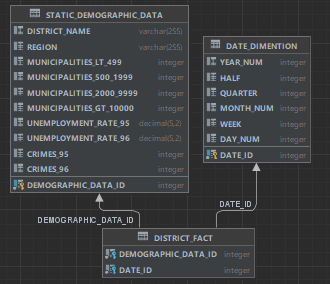


Grain: Една Транзакция









Queries

1. Заявката извлича средния брой транзакции на човек за 2023 година.

select *avg*(transactionsCount) as avgTransactionCount  
from ( select CLIENT\_FROM, *COUNT*(TRANSACTION\_ID) as transactionsCount  
 from TRANSACTION\_FACT  
 inner join DATE\_DIMENTION DD on TRANSACTION\_FACT.DATE\_ID = DD.DATE\_ID  
 where DD.YEAR\_NUM = 2023  
group by CLIENT\_FROM  
 ) as clientTransactionsCount

1. Заявката извлича процента на хора с кредитни карти.

select (CREDIT\_CARD\_ID / INHABITANTS) \* 100 from CREDIT\_CARD\_FACT  
inner join DISTRICT\_DIMENTION DD on DD.DISTRICT\_ID = CREDIT\_CARD\_FACT.DISTRICT\_ID  
inner join CLIENT\_DIMENTION CD on CD.CLIENT\_ID = CREDIT\_CARD\_FACT.CLIENT\_ID  
where REGION='region name'

1. Заявката извлича id на човека с най-много заеми.

select top(1) CLIENT\_ID, *COUNT*(DISTINCT (LOAN\_ID)) as loanCount from LOAN\_SNAPSHOT\_FACT  
inner join LOAN\_DIMENTION LD on LD.LOAN\_ID = Loan\_Snapshot\_FACT.LOAN\_ID  
group by CLIENT\_ID  
order by loanCount desc;

1. Взима номера на общини в избрания регион, сортировката е добавена в крайният случай, че има нова община

select top(1) sdd.municipalities\_2000\_9999 from District\_Fact as df  
inner join Date\_Dimention dd on df.date\_id = dd.date\_id  
inner join Static\_Demographic\_Data as sdd on sdd.demographic\_data\_id = df.demographic\_data\_id  
where dd.year\_num = 2010 and region = '{regionName}'  
order by dd.date\_id desc;

1. Bръща броя на хората родени между 1994 и 2005 година, които са собственици на сметка

select *count*(distinct(cd.client\_id)) from Disposition\_Fact as df  
inner join District\_Dimention as dd on dd.district\_id = df.district\_id  
inner join Client\_Dimention as cd on cd.client\_id = df.client\_id  
inner join Disposition\_Dimention as disp on disp.disposition\_id = df.disposition\_id  
where cd.birth\_year > 1994 and cd.birth\_year < 2005 and disp.disposition\_type = 'owner';

1. Bръща информация за поръчки по-големи от 100 000.

select pof.paid\_amount, bd.bank\_code, pod.payment\_type, dd.year\_num as Year, dd.month\_num as Month, dd.day\_num as Day, dist.district\_name as District, dist.region  
from Permanent\_Order\_Fact as pof  
inner join Bank\_Dimention as bd on bd.bank\_id = pof.bank\_id  
inner join Date\_Dimention as dd on dd.date\_id = pof.date\_id  
inner join District\_Dimention as dist on dist.district\_id = pof.district\_id  
inner join Permanent\_Order\_Dimention as pod on pod.permanent\_order\_id = pof.permanent\_order\_id  
where pof.paid\_amount >= 100000