Софийски Университет „Св. Климент Охридски“
Факултет по Математика и Информатика

Курсова работа по:
Складове от данни и бизнес анализ

Тема: Създаване на Data Warehouse

Изготвен от: Павел Романов, Йоан Пачовски,
Габриел Миндев и Трендафил Дъбов

Съдържание

Описание на задачата

Този проект има за цел да покаже как се създава Data Warehouse – централизирано място за съхранение на данни, съдържащо както текуща, така и historical информация, която се обработва и анализира с цел взимането на точни и добре премислени бизнес решения.

В документацията ще опишем защо избрахме подходът на Kimbell за построяването на складовете от данни, както и ETL процесът, който използвахме за взимането, трансформирането и зареждането на данните от първоначалния източник.

Също така ще покажем различните схеми, които направихме при създаването на склада от данни. Всяка една от тях ще бъде описана подробно, за да може да е ясен процесът на работа, който сме използвали.

Накрая ще демонстрираме няколко заявки, както и как могат те да се тълкуват за да ни дадат необходимата информация за взимането на необходимите решения за развитието на един бизнес.

Защо избрахме метода на Kimbell ?

При този модел data marts се създават първи в зависимост от бизнес изискванията, които имаме.

Следващата стъпка е главните източници на информация да бъдат оценени и чрез използването на ETL процеса да се вземе информацията от тях и се постави в staging area-та. След като вече е качена там може да се премине към създаването на дименционния модел. В него имаме:

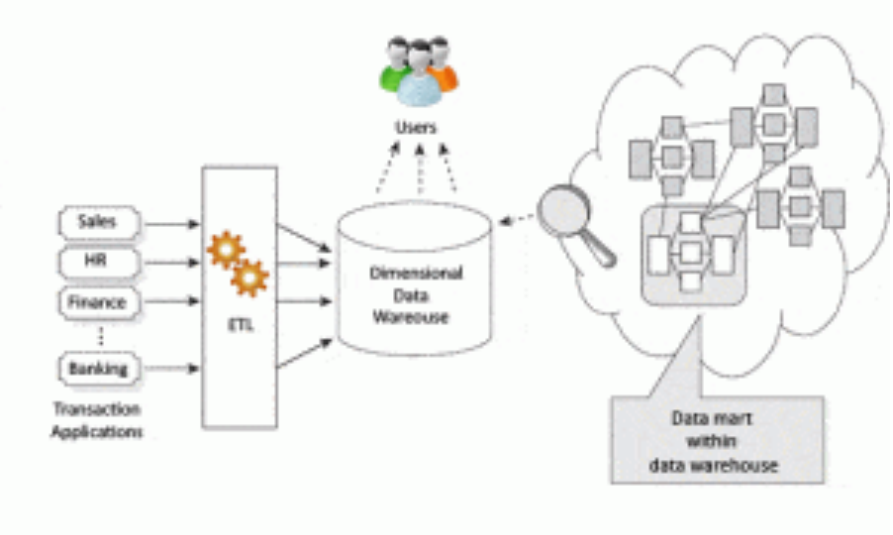
- Факт таблици, които съдържат числена информация, даваща ни важна бизнес ориентирана информация

- Дименционни таблици, носещи контекст за съответните факти.

Фундаментален елемент за създаването на Data warehouse е Star Schema-та. При нея всяка факт таблица е свързана с дименционни таблици. Моделът на Кимбъл ни позволява да изградим няколко като всяка от тях ни дава необходимата информация за всяка една от нуждите на бизнеса.

Избрахме този модел, защото:

- При него се използва Star schema. Тя има лесна за разбиране денормализирана структура, която улеснява правенето на заявки и техния анализ.
- По-лесен е за имплементация и поддръжка от метода на Inmon



Модел на Kimbell

Описание на ETL процеса

Екстракция – взимаме информацията от първоначалните източници. Тя първоначално е в raw формат и трябва да се обработи за да отговаря на необходимия формат за създаването на складовете от данни.

Трансформация – данните биват обработени по следния начин:

- Ако имаме колони даващи ни еднаква информация, но съхранени в различни формати, то ние избираме общ стандарт за тяхното съхранение.

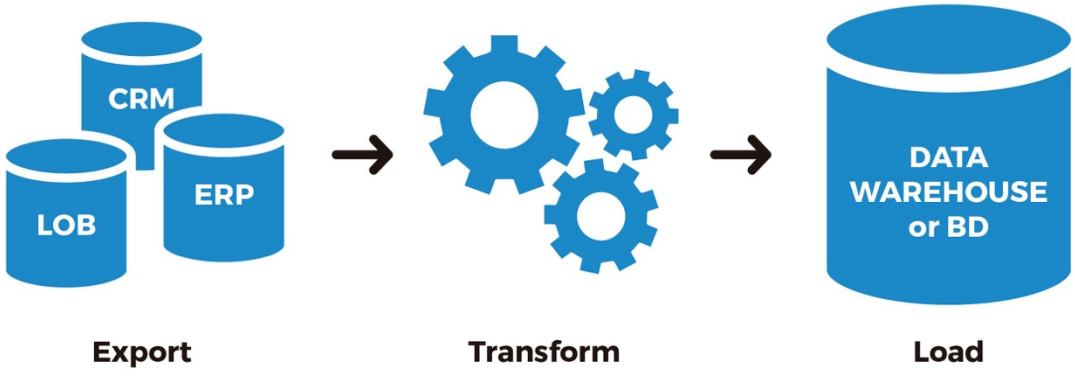
  Пример: Ако датата в една от колоните се съхранява във формат varchar(6) – 980320, а в друга колона 20/03/1998, то ние ще искаме данните да се съхраняват или в единия формат, или в другия.

- Премахваме еднаквите редове в таблиците, ако има такива.

- Колоните, даващи ни еднаква информация да бъдат с еднаква дължина и съхранявани в еднакъв data формат – varchar, number, т.н.

- Ако информация в дадена колона е на чужд език (както в нашия случай на чешки) ние трябва да променим стойностите с такива на разбираем език (английски например)

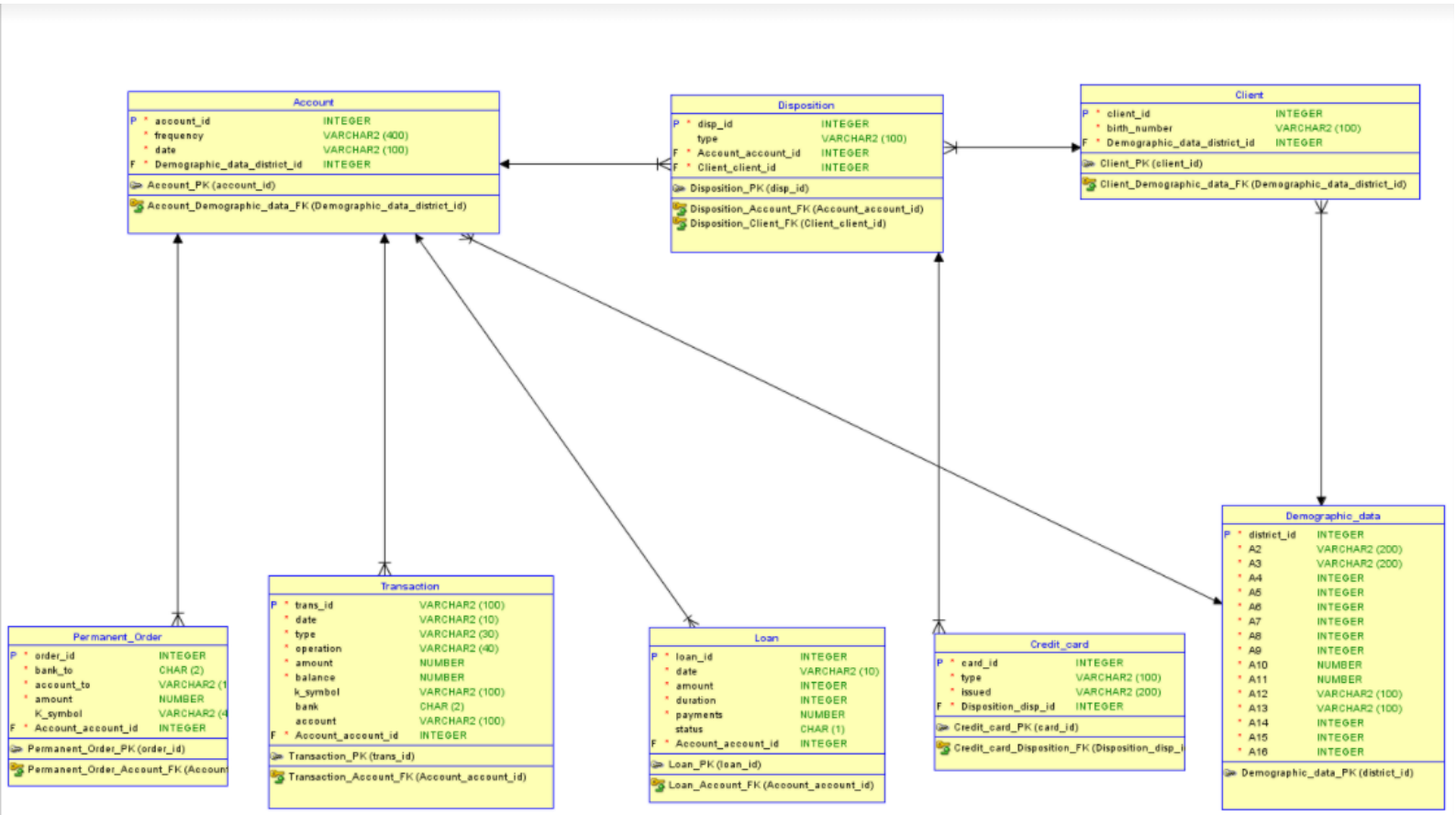
Зареждане – поставяме вече трансформираните данни в staging area-та, откъде после ще бъдат прехвърлени в таблиците на дименционния модел.



Source data model

Въпреки, че имаме първоначална входна информация, съдържаща използваните таблици и техните атрибути, ние трябва да изградим техния модел за да можем да разберем и визуализираме връзките между тях, както и да поставим съответните primary и foreign ключове.

Без тази информация няма да можем да създадем и правилните модели, необходими за създаването на складовете от данни.



Моделът съдържа следните таблици:

Account – дава ни информация за съответния профил и съдържа следните атрибути:

- Account_id – това е нейния primary key
- District_id_– чужд ключ за връзката с таблицата demographic_data, като връзката с нея е много към едно.
- Date – кога е създаден профила
- Frequency – честота на изискване на изявления. Приема стойности:
    - Monthly – месечно изискване
    - Weekly – седмично изискване
    - After - изискване след направена транзакция

Permanent_order – описва направените поръчки. Има следните атрибути:

- Order_id - това е нейния primary key
- Account_id - чужд ключ за връзката с таблицата account, като връзката с нея е много към едно.
- Bank_to_– банката на получателя.
- Account_to_– акаунт на получателя.
- Amount – сума на поръчката.
- k_symbol_– характеризация на плащането. Приема стойности:
    - insurance_– застраховка
    - household_– household payment
    - leasing_– лизингово плащане
    - loan_– заем

Transaction – описва транзакциите на акаунти. Има следните атрибути:

- trans_id – това е нейния primary key
- account_id – чужд ключ за връзката с таблицата account, като връзката с нея е много към едно.
- date – дата на транзакцията

- type – тип на транзакцията
    - credit - кредит
    - withdrawal - теглене
- operation – метод на транзакцията
    - credit card withdrawal – чрез кредитна карта
    - credit in cash – кредит в кеш
    - collection from another bank – теглене от друга банка
    - withdrawal in cash – теглене в кеш
    - remittance to another bank – пращане на сума в друга банка
- amount – сума изтеглени / изпратени пари
- balance – баланс след транзакцията
- k_symbol – характеристика на транзакцията
    - insurance_– застраховка
    - household_– household payment
    - loan_– заем
    - interest – теглене с лихва
    - sanction – наказателна лихва при отрицателен баланс
    - pension – пенсия
    - statement – плащане по изявление
- bank – банка на партньора
- account – акаунт на партньора

Client – дава характеристика на клиентите. Има следните атрибути:

- client_id – това е нейния primary key
- birth number – пол и рождена дата на клиента.
- district_id – адрес на клиента. Това е чужд ключ за връзката с таблицата demographic_data много към едно.

Disposition – свързва account и client. Има следните атрибути:

- disp_id - това е нейния primary key

- client_id - чужд ключ за връзката с таблицата client, като връзката с нея е много към едно.
- account_id - чужд ключ за връзката с таблицата account, като връзката с нея е много към едно.
- Type – вид на disposition-а.

Loan – описва заем даден на даден акаунт. Има следните атрибути:

- Load_id - това е нейния primary key
- Account_id - чужд ключ за връзката с таблицата account, като връзката с нея е много към едно.
- Date – дата на даването на заема
- Amount – сума на заема
- Duration – продължителност на заема
- Payments – месечни вноски по заема
- Status – статут на плащането на заема
    - A – плащането е извършено
    - B – договорът е завършен, заемът не е платен
    - C – договорът не е завършен, до момента се плаща
    - D – договорът не е завършен. Акаунта е в дълг

Опциите за атрибут status в staging area-та ще бъдат променени

Credit_card – описва кредитна карта, издадена на съответния акаунт. Има следните атрибути

- Card_id - това е нейния primary key
- Disp_id - чужд ключ за връзката с таблицата disposition, като връзката с нея е много към едно.
- Type – вид кредитна карта
    - Junior
    - Classic
    - Gold
- Issued – дата на издаване

Demographic_data – описва характеристиките на даден квартал / регион. Има следните атрибути:

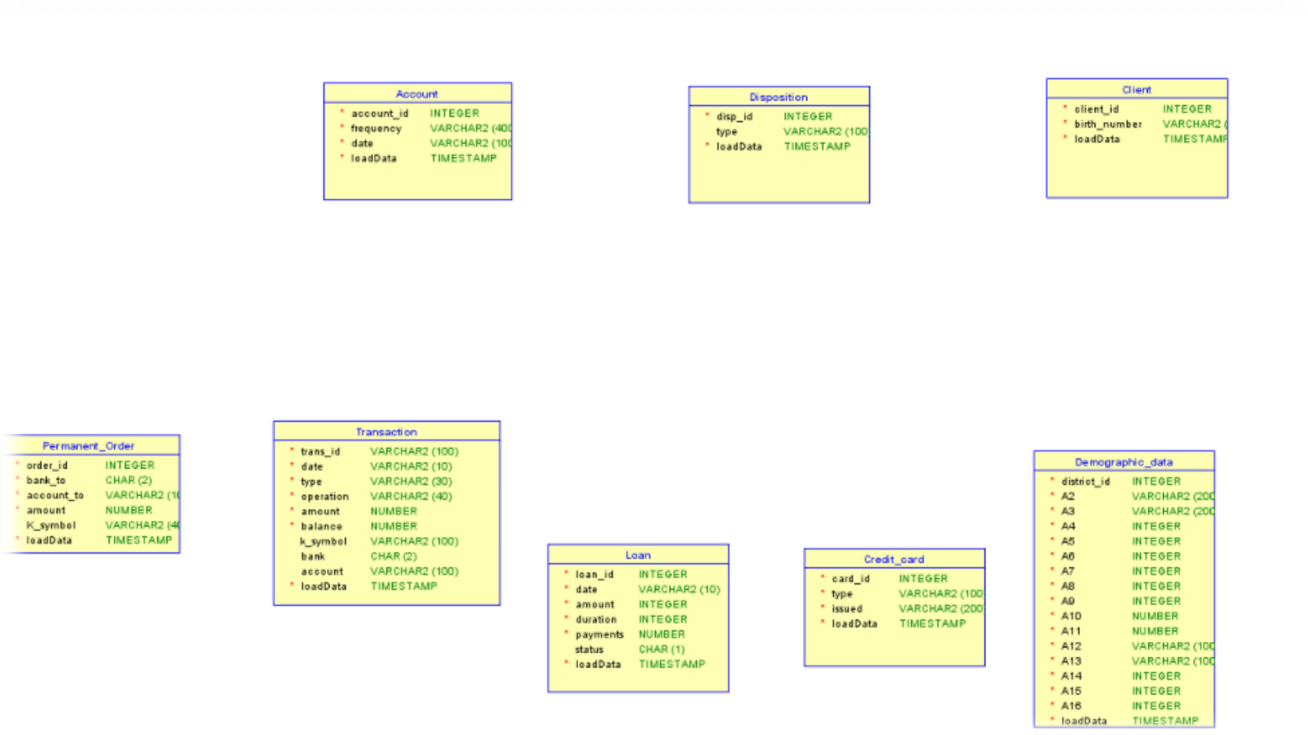
- A1 = district_id - това е нейния primary key
- A2 – име на квартал
- A3 - регион
- A4 – брой жители
- A5 – брой общини с жители < 499
- A6 – брой общини с жители 500-1999
- A7 – брой общини с жители 2000-9999
- A8 – брой общини с жители > 10000
- A9 – брой градове
- A10 – съотношение градски жители и селски
- A11 – средна заплата
- A12 – безработица 1995
- A13 – безработица 1996
- A14 – брой бизнесмени за 1000 жители
- A15 – брой престъпления 1995
- A16 – брой престъпления 1996

След като сме направили модела, можем да прехвърлим данните към staging area-та, където те да могат да бъдат трансформирани преди да преминат към Data Warehouse-а.

Staging area Data Model

Staging area-та е мястото, където данните се съхраняват преди да бъдат заредени в складовете от данни. Тук също се и изпълнява трансформацията им за да отговарят на необходимия стандарт и да могат да бъдат използвани за бъдещи заявки и анализ на данните.

Staging area-та има следния модел:



При него сме махнали чуждите ключове и връзките между отделните таблици за да може да става по-лесно зареждането на данните в складовете от данни.

Направили сме и следните трансформации:

- В таблицата client в условието е казано, че при определени клиенти месеците при birth_number са във формат YYMM+50DD. Вече всички дати в тази таблица са от формата YYMMDD.

- Голяма част от таблиците имаха атрибути със стойности на чешки език. След трансформацията вече всички на английски език, като техните стойности и обяснения са описани в горната точка.

- При създаването на дименционния модел вече имаме таблица calendar, при която датите от предишен формат YYMMDD са разделени в колони съответно (година, месец, ден)

- В таблицата loan с атрибут status сме променили данните по следния начин:
    - A-> contract finished, no problems
    - B-> contract finished, loan no paid
    - C-> running contract, OK so far
    - D-> running contract, client in debt

Data Warehouse model

В тази част преминаваме към създаването на дименционния модел. Първо трябва да определим тези атрибути, които ни дават важна информация за бизнеса и могат да бъдат използвани в заявките и анализа на данните. Чрез тези атрибути можем да създадем факт таблиците и съответните дименционни таблици, които ни дават контекст за тях.

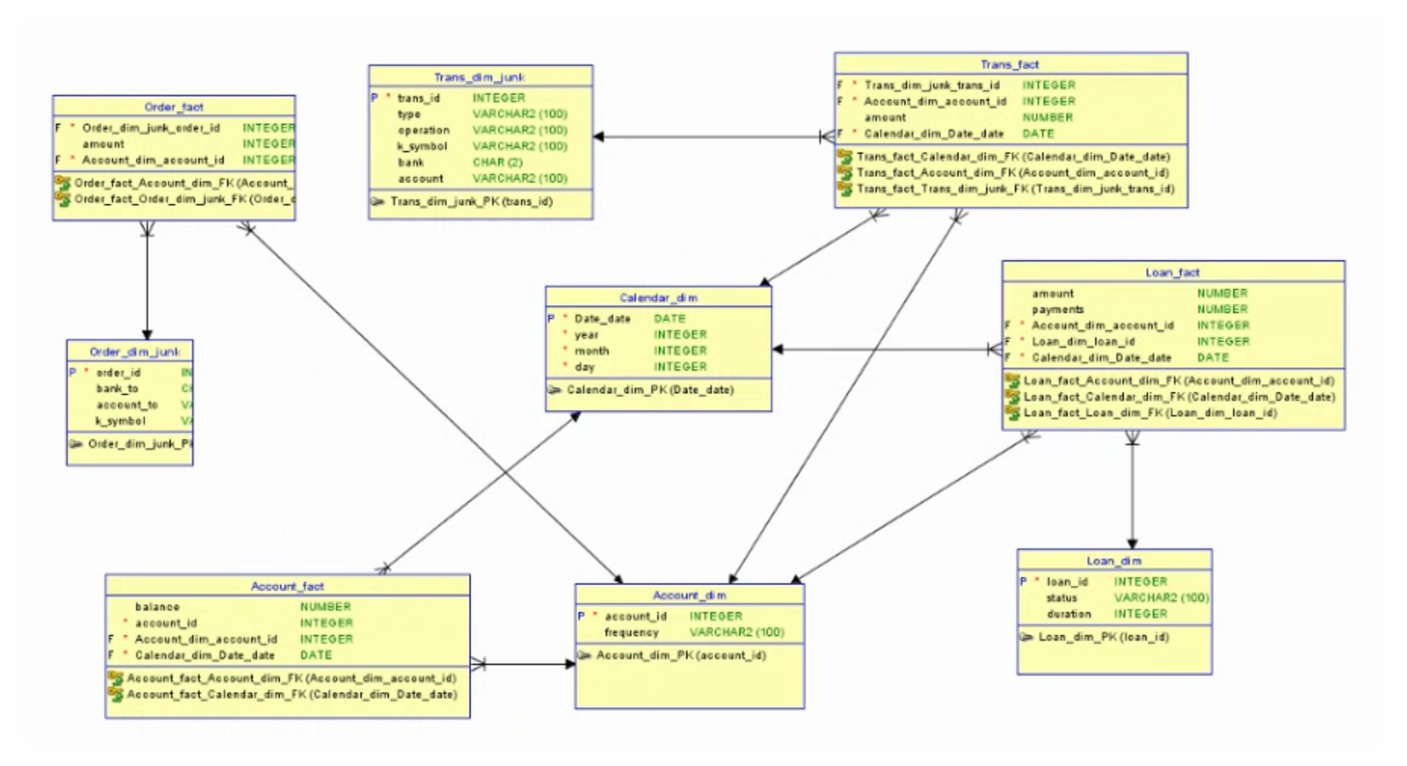
Атрибутите, които избрахме са:

- Amount в order таблицата, която ни дава информация за сумата при правенето на поръчка.

- Amount от transaction таблицата, които ни дават информация за сумата при направена транзакция

- Amount и payments от loan таблицата, които ни дават информация размера на заема и размера на месечните плащания по него.

След като се взели в предвид тези данни моделът ни изглежда така:



Създали сме три факт таблици:

- **Order_fact** – тя има следните атрибути
  - Order_dim_order_id – това е чужд ключ за връзката с дименцията order_dim_junk
  - Account_dim_account_id - това е чужд ключ за връзката с дименцията account_dim
  - Amount – дава ни информация за стойността на поръчката

- **Loan_fact** – тя има следните атрибути
  - Loan_dim_loan_id – чужд ключ за връзката с дименцията loan_dim
  - Calendar_dim_Date_date – чужд ключ за връзката с дименцията Calendar_dim
  - Account_dim_account_id - това е чужд ключ за връзката с дименцията account_dim
  - Amount – сума на заема
  - Payments – месечна вноска

- **Trans_fact** – тя има следните атрибути
    - Trans_dim_trans_id - чужд ключ за връзката с дименцията trans_dim_junk
    - Calendar_dim_Date_date – чужд ключ за връзката с дименцията calendar_dim
    - Account_dim_account_id - това е чужд ключ за връзката с дименцията account_dim
    - Amount – сума на транзакция

Всички връзки между факт таблици и дименционни е много към едно!

Сега ще опишем и създадените дименционни таблици:

Order_dim_junk

- Order_id – primary key на таблицата
- Bank_to – банка получател на поръчка
- Account_to – акаунт получател
- K_symbol – характеристика на плащането

Account_dim

- Account_id - primary key на таблицата
- Frequency - честота на изискване на изявления

Trans_dim_junk

- Trans_id - primary key на таблицата
- Type – тип на транзакцията
- Operation – метод на транзакцията
- K_symbol – характеристика на транзакцията
- Bank – банка на партньора
- Account – акаунт на партньора

Calendar_dim – това е новата таблица, която създадохме за да се разделим досегашния формат на датите (YYMMDD) в три различни атрибута (година, месец, ден)

- Year - година
- Month - месец
- Day - ден
- Date_date – primary key дименцията

Loan_dim

- Load_id – primary key на таблицата
- Status – статус за плащането на заема
- Duration – продължителност за изплащане на заема

След като вече сме създали dimensional model-а зареждаме трансформираните данни в него и вече можем да правим заявки с цел business intelligence

Полезни заявки

Заявка 1. Първата заявка ни дава информация за потребителите, които са задължнели на банката като връща тези, които са изтеглили най-голяма обща сума от заеми и колко месеца им оставят до изплащането им. Подредени са в намаляващ ред и са върнати първите 5 потребителя.

```
SELECT account_id,sum(amount),duration from DW_LOAN_FACT
join dw_loan_dim on dw_loan_fact.loan_dim_loan_dim_id= dw_loan_dim.loan_id
join dw_account_dim on dw_loan_fact.account_dim_account_id= dw_account_dim.account_id
where dw_loan_dim.status = 'running contract, client in debt'
group by account_id, duration order by sum(amount) desc
fetch first 5 rows only;
```

Резултат:

	ACCOUNT_ID	SUM(AMOUNT)	DURATION
1	2335	541200	60
2	10451	482940	60
3	7966	473280	60
4	4794	465504	48
5	3711	460980	60

Заявка 2. Втората заявка ни дава потребителите, получили пенсионни транзакции като са подредени в намаляващ ред по общата сума направени транзакции за всеки един от тях.

```
select account_dim_account_id as "User", sum(amount) as "Total Sum"
from dw_trans_fact
join dw_trans_dim on dw_trans_fact.trans_dim_trans_id = dw_trans_dim.trans_id
where k_symbol= 'pension'
group by account_dim_account_id order by sum(amount) desc;
```

Резултат:

	User	Total Sum
1	165	459888
2	2293	451915
3	163	448766
4	428	443138
5	2712	442468
6	1784	441276
7	300	441088
8	112	440994
9	1401	438400
10	2357	437512
11	1775	436716
12	1782	431772
13	1671	426855
14	519	426207
15	3691	425100
16	616	424452
17	1930	423800
18	3008	422974
19	1222	422770
20	2849	420651
21	2618	419776
22	3243	419253
23	1947	418460
24	3604	416700
25	576	415869

Заявка 3. Третата заявка ни дава информация за потребителите, направили най-големи тегления с кредитни карти за първото тримесечие на 1998 година. Подредени са по общата сума в намаляващ ред.

```
select account_dim_account_id as "User", sum(amount)as "Total Sum" from dw_trans_fact
join dw_trans_dim on dw_trans_fact.trans_dim_trans_id = dw_trans_dim.trans_id
where calendar_dim_year = 98 and calendar_dim_month > 0  and calendar_dim_month < 4 and operation='credit card withdrawal'
group by account_dim_account_id order by sum(amount) desc;
```

Резултат:

	User	Total Sum
1	2242	30000
2	1485	29700
3	1519	28800
4	1203	28300
5	1750	28100
6	3654	27300
7	904	26200
8	10520	25500
9	73	25200
10	456	24600
11	3476	23200
12	1780	23000
13	1408	22900
14	1016	22700
15	886	22300
16	1151	21800
17	68	21500
18	1734	21300
19	2701	21300
20	1919	20500
21	1112	20300
22	4343	20300
23	2034	20100
24	2219	20100

Заявка 4. Дава ни информация за 10-те банки получили най-големи суми при правене на household поръчки. Те са подредени в намаляващ ред по общата сума получени поръчки

```
select bank_to as "BANK",sum(amount)as "SUM RECEIVED"
from dw_order_fact
join dw_order_dim on dw_order_fact.order_dim_order_id=dw_order_dim.order_id
where k_symbol='HOUSEHOLD'
group by  bank_to order by sum(amount) desc
fetch first 10 rows only;
```

Резултат:

	BANK	SUM RECEIVED
1	ST	1176435
2	KL	1168704
3	EF	1165426
4	IJ	1125738
5	QR	1111723
6	AB	1111015
7	WX	1088112
8	YZ	1072609
9	UV	1053848
10	GH	1038683