**Solution concept – Business background**

As an example of a business idea, consider the work of the Belarusian chain of restaurants of Georgian cuisine.

**Brief overview of the chain of restaurants Khinkalnya:**

Khinkalnya is a family, they cherish the recipes of their favorite dishes and cook only the most delicious: khinkali with fragrant broth and juicy filling, khachapuri boats, warming kharcho and tender chikhirtma, spicy kebab and Georgian pkhali. And the most delicious homemade Napoleon. In Khinkalne, as at home, you can celebrate the holiday with sparkling Bedagoni and enjoy an ordinary day with a glass of homemade Kakhetian wine. Gather the whole family at a large table or make an appointment with a dear person, or just drop in for a delicious lunch. Everyone is comfortable in Khinkaln: there is a children’s menu and comfortable chairs, as well as entertainment for little guests.

Let this chain of restaurants face certain difficulties related to incorrect data handling. **Some of the possible problems:**

1. lack of business intelligence from multiple sources;

2. reduced query and system performance;

3. lack of timely access to data;

4. the lack of historical intelligence.

We offer the customer to use DWH(we will consider 2 types of storage schemes: star scheme and snowflake scheme) **for the following reasons:**

**1. Access to the required data.** Since the chain is quite large, you need to collect permissions and accesses to receive data from different sources. Each department in such a situation, as a rule, has its own databases with its own passwords, which will need to be requested separately. At DWH, everything you need will already be at hand in finished form. You can just go and pull the necessary statistics there.

**2. Keeping the right data.** Data in DWH is not lost and is stored in a form convenient for decision making: there are historical records, there are aggregated values. The operational database may not have this information.

**3. Sustainability of business systems.** DWH is optimized for the work of analysts, and these guys can request very large amounts of information. If they do this with the help of DWH, it's okay, even if their request will be processed for a very long time. And if you request too many records from a regular server database, it can go into failure before the end of the request from analytics and create problems for other systems. DWH eliminates the risk of analysts hanging or breaking something.

**Business requirements:**

1. chain has several restaurants in different countries. Each restaurant has its own geographical location. The warehouse must take this fact into account;

2. calculation of statistics of ordered dishes through delivery monthly (you can collect information for large time periods due to the time hierarchy);

3. statistics of ordered dishes for different periods of time (during the day, during the week, during the season of the year);

4. calculation of information about visits to restaurants in different regions, cities or even districts of the same city.

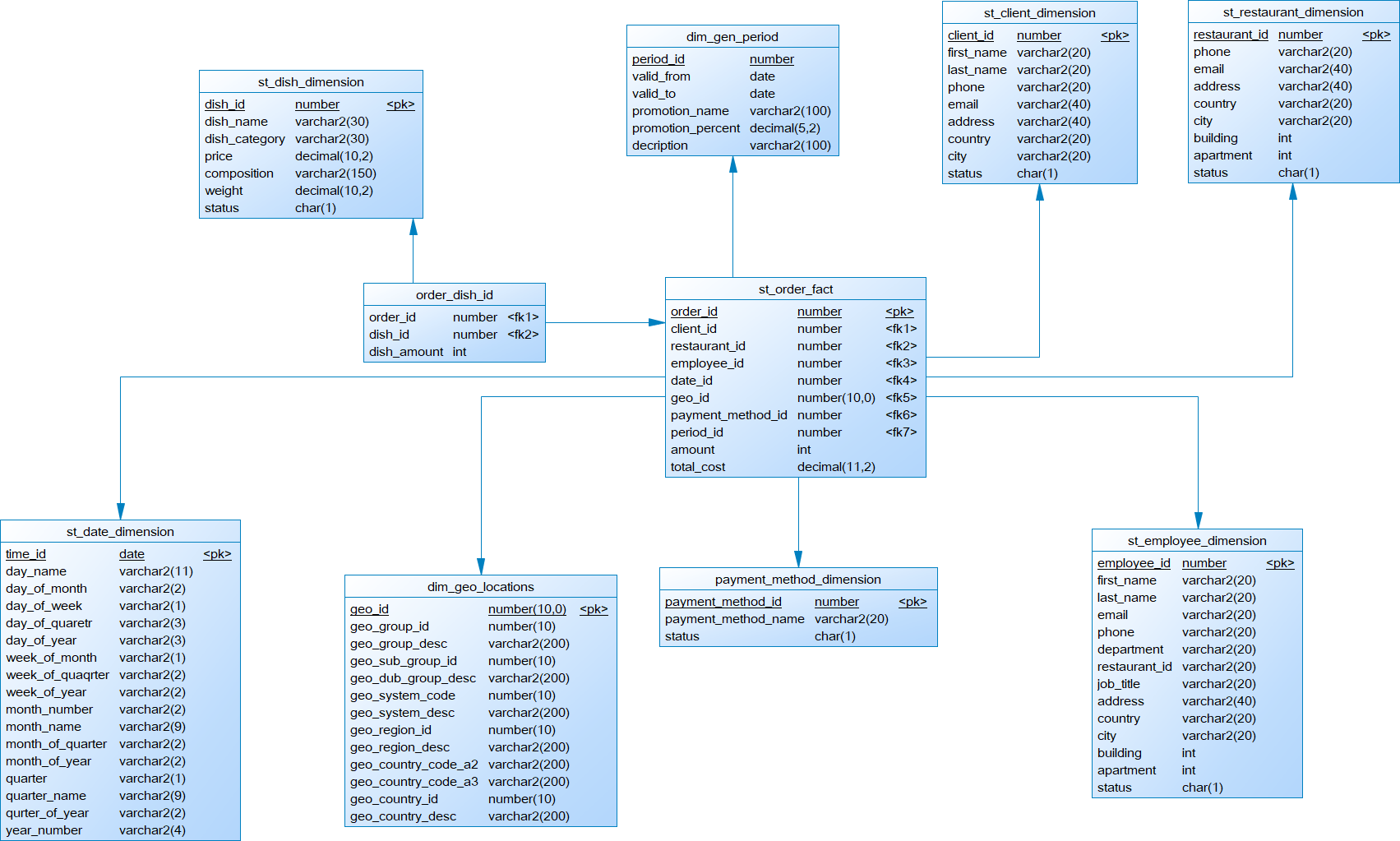
**Technical requirements:**

1. keep info from the beginning of business;
2. persistent and observable storage access;
3. ability to process large amounts of information per day (100 thousands

rows);

1. high performance and high availability;
2. all the information must be protected.

**Star-Scheme physical diagram**

****

**Dimensions Types**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Size | DW – Merged Dimension | Description |
| DIM\_GEN\_TIMES | SCD1 | BIG | DW.T\_DAY  DW.T\_WEEKS  DW.T\_MONTHS  DW.T\_QUARTERS  DW.T\_YEARS | TBD – Example row |
| DIM\_GEO\_LOCATIONS | SCD1 | SMALL | GEO\_ID  GEO\_GROUP\_ID  GEO\_GROUP\_DESC  GEO\_SUB\_GROUP\_ID  GEO\_DUB\_GROUP\_DESC  GEO\_SYSTEM\_CODE  GEO\_SYSTEM\_DESC  GEO\_REGION\_ID  GEO\_REGION\_DESC  GEO\_COUNTRY\_CODE\_A2  GEO\_COUNTRY\_CODE\_A3  GEO\_COUNTRY\_ID  GEO\_COUNTRY\_DESC | This kind of dimension contains information about all countries, subregions, regions of the world where the company's stores are located. And also enters information on the types of economic development and unions according to the international classification. |
| CLIENT\_DIMENSION | SCD1 | BIG | CLIENT\_ID  FIRST\_NAME  LAST\_NAME  PNONE  EMAIL  ADDRESS  COUNTRY  CITY  STATUS | This kind of dimension contains detailed information about clients. |
| DISH\_DIMENSION | SCD2 | BIG | DISH\_ID  DISH\_NAME  DISH\_CATEGORY  PRICE  COMPOSITION  WEIGHT  STATUS | This kind of dimension contains detailed information about the restaurant’s dishes, including the name of dish, category, composition and weight. To do so, provided the opportunity for dimension Type SCD 2 perfectly partitions history because each detailed version of a dimensional entity is correctly connected to the span of fact table records for which that version is exactly correct. |
| RESTAURANT\_DIMENSION | SCD1 | SMALL | RESTAURANT\_ID  PHONE  EMAIL  ADDRESS  COUNTRY  CITY  BUILDING  APARTMENT  STATUS | This kind of dimension contains detailed information about restaurant including the restaurant’s address, email and phone. |
| EMPLOYEE\_DIMENSION | SCD2 | BIG | EMPLOYEE\_ID  FIRST\_NAME  LAST\_NAME  DATE\_OF\_BIRTH  EMAIL  PHONE  DEPARTMENT  RESTAURANT\_ID  JOB\_TITLE  ADDRESS  COUNTRY  CITY  BUILDING  APARTMENT  STATUS | This kind of dimension contains detailed information about employee. To do so, provided the opportunity for dimension Type SCD 2 perfectly partitions history because each detailed version of a dimensional entity is correctly connected to the span of fact table records for which that version is exactly correct. |
| PAYMENT\_METHOD\_DIMENSION | SCD1 | SMALL | PAYMENT\_METHOD\_ID  PAYMENT\_METHOD \_NAME  STATUS | This kind of dimension contains information about the payment method used. |
| DIM\_GEN\_PERIOD | SCD2 | BIG | PERIOD\_ID  VALID\_FROM  VALID\_TO  PROMOTION\_NAME  PROMOTION\_PERCENT  DECRIPTION | A specific type of dimension that allows grouping facts based on logic (the duration of dish discounts). |
| DATE\_DIMENSION | SCD1 | BIG | TIME\_ID  DAY\_NAME  DAY\_OF\_MONTH  DAY\_OF\_WEEK  DAY\_OF\_QUARTER  DAY\_OF\_YEAR  WEEK\_OF\_MONTH  WEEK\_OF\_QUARTER  WEEK\_OF\_YEAR  MONTH\_NUMBER  MONTH\_OF\_QUARTER  MONTH\_OF\_YEAR  QUARTER  QUARTER\_NAME  QUARTER\_OF\_YEAR  YEAR\_NUMBER | This kind of information contains information about days, weeks, months, quarters and years. |

**Dimensions Hierarchies**

**DATE\_DIMENSION**

**Hierarchy DAY-WEEK-MONTH-YEAR**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| DAY | DAY | Store all days at the week | DAY\_OF\_WEEK |
| WEEK | WEEK | Store all weeks at the month | WEEK\_OF\_MONTH |
| MONTH | MONTH | Store all months at the year | MONTH\_OF\_YEAR |
| YEAR | YEAR | Store all years | YEAR\_NUMBER |

**Hierarchy DAY-MONTH-QUARTER-YEAR**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| DAY | DAY | Store all days at the month | DAY\_OF\_MONTH |
| MONTH | MONTH | Store all months at the quarter | MONTH\_OF\_QUARTER |
| QUARTER | QUARTER | Store all quarters at the year | QUARTER\_OF\_YEAR |
| YEAR | YEAR | Store all years | YEAR\_NUMBER |

**Hierarchy DAY-QUARTER-YEAR**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| DAY | DAY | Store all days at the quarter | DAY\_OF\_QUARTER |
| QUARTER | QUARTER | Store all quarters at the year | QUARTER\_OF\_YEAR |
| YEAR | YEAR | Store all years | YEAR\_NUMBER |

**Hierarchy DAY-WEEKS-YEAR**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| DAY | DAY | Store all days at the quarter | DAY\_OF\_WEEK |
| WEEKS | WEEKS | Store all weeks at the year | WEEK\_OF\_YEAR |
| YEAR | YEAR | Store all years | YEAR\_NUMBER |

**GEO\_LOCATIONS\_DIMENSION**

**Hierarchy COUNTRY – REGION – GEO\_GROUP – GEO\_SUB\_GROUP**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| COUNTRY | GEO\_COUNTRY | Store all countries for each region | GEO\_COUNTRY\_ID |
| REGION | GEO\_REGION | Store all regions for each geo\_group | GEO\_REGION\_ID |
| GEO\_GROUP | GEO\_GROUP | Store all geo\_groups for each geo\_sub\_group | GEO\_GROUP\_ID |
| GEO\_SUB\_GROUP | GEO\_SUB\_GROUP | Store all geo\_sub\_groups | GEO\_SUB\_GROUP\_ID |

**DISH\_DIMENSION**

**Hierarchy NAME – CATEGORY**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| NAME | NAME | Store all dishes for each category | DISH\_ID |
| CATEGORY | CATEGORY | Store all categories | CATEGORY\_ID |

**ENPLOYEE\_DIMENSION**

**Hierarchy EMPLOYEE – DEPARTMENT**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| EMPLOYEE | EMPLOYEE | Store all employees for each department | EMPLOYEE\_ID |
| DEPARTMENT | DEPARTMENT | Store all departments | DEPARTMENT \_ID |

**Hierarchy JOB\_TITLE – DEPARTMENT**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| JOB\_TITLE | JOB\_TITLE | Store all job\_titles for each department | JOB\_TITLE\_ID |
| DEPARTMENT | DEPARTMENT | Store all departments | DEPARTMENT \_ID |

**Facts Aggregations**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Code | Table Name | Additive | Description |
| Total amount of paid orders | ORDERS\_AMOUNT | ORDER\_FACT | + | Calculate total amount of orders in the selected period or restaurant, city, country and atc or pay method or dilevery (profit) |
| Total number of paid orders | ORDERS\_NUMBER | ORDER\_FACT | + | Calculate total amount of orders in the selected period or restaurant, city, country and atc or pay method or dilevery ( visit statistics) |
| Quantity of each paid dish | DISH\_QUANTITY | ORDER\_FACT | + | Calculate quantity of each paid dish in the selected period or restaurant, city, country and atc or pay method or dilevery |
| Quantity of paid dishes in each category | DISH\_QUANTITY\_IN\_CATEGORY | ORDER\_FACT | + | Calculate quantity of paid dishes in each category in the selected period or restaurant, city, country and atc or pay method or dilevery |
| Average order amount | AVG\_ORDER\_AMOUNT | ORDER\_FACT | - | Calculate average order amount in each category in the selected period or restaurant, city, country and atc or pay method or dilevery |

**Schema of simple Data Warehouse Architecture**

|  |  |  |  |
| --- | --- | --- | --- |
| Level Type | Object Name | Tablespace | Description |
| Storage Level  SA\_\* | SA\_CLIENTS | ts\_sa\_clients\_data\_01 | Loading from structured files (exp. XML\*). Contains information about client. |
| SA\_ORDERS | ts\_sa\_orders\_data\_01 | Loading from float files. Contains information about order. |
| SA\_EMPLOYEES | ts\_sa\_employees\_data\_01 | Loading from structured files (exp. XML\*). Contains information about employee. |
| DW – Cleansing level | DW\_CL | ts\_dw\_cl | Loading from stage level system. Contains all information prepare it for further usage (cleaning it). |
| DW – Level | DW\_DATA | ts\_dw\_data\_01 | LOADING data from cleansing system. Contains clean information tending to the 3rd normal form ready for preparing star schema. |
| DW – Prepare Star Cleansing Level | SAL\_DW\_CL | ts\_dw\_str\_cls | LOADING data from DW system. Contains views merging objects from DW level. |
| STAR - Cleansing | SAL\_CL | ts\_sal\_cl | LOADING data from DW\_CL system. Contains views from previous level but clean any redundancy. |
| STAR - Level | DM\_CLIENT | ts\_sa\_dim\_client\_01 | Store information about dim client. |
| DM\_ORDER | ts\_sa\_fct\_order\_01 | Store information about fact order. |
| DM\_EMPLOYEE | ts\_sa\_dim\_employee\_01 | Store information about dim employee. |
| DM\_RESTAURANT | ts\_sa\_dim\_restaurant\_01 | Store information about dim restaurant. |
| DM\_DISH | ts\_sa\_dim\_dish\_01 | Store information about dim dish. |
| DM\_PAYMENT\_METHOD | ts\_sa\_dim\_payment\_method\_01 | Store information about dim payment method. |

**Partitioning Facts**

The orders are divided by quarters using range partitioning by DATE\_ID. The quarters are divided using hash partitioning by CLIENT\_ID.

Example of a result table of facts with partition taken into account:

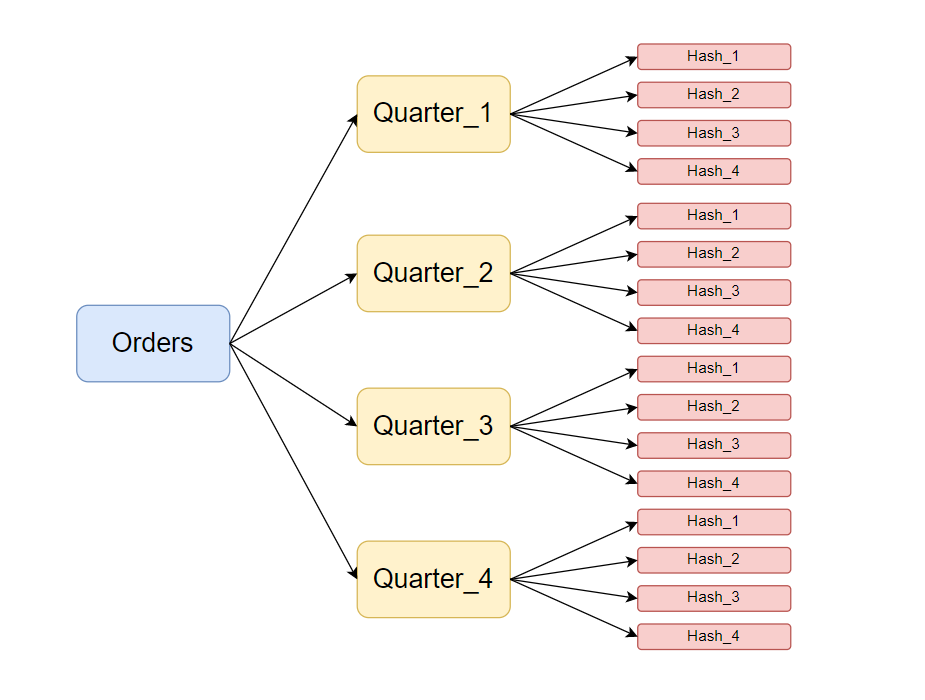
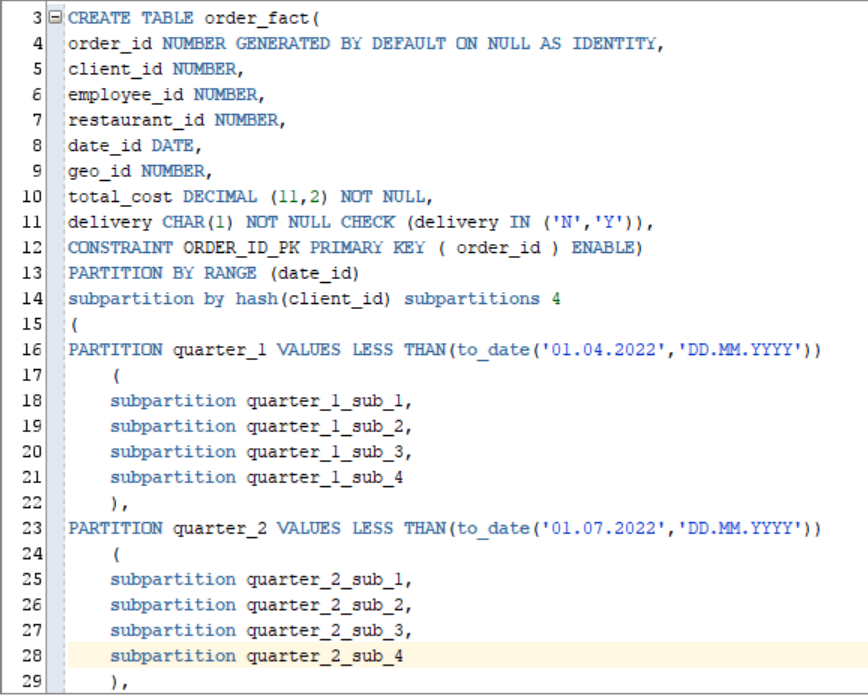
****

Table with the suggested partition option:

****

**Strategy of Parallel execution**

The use of parallelism can be a good example of increasing the speed and efficiency of a data warehouse.

Since I have tables in my data warehouse that need to be constantly updated or new data added, such as ORDER\_FACT, CLIENT\_DIMENSION, EMPLOYEE\_DIMENSION, it makes more sense to use the DML parallel to run this process more efficiently at the DW, CL, and SA levels.