SAPIENZA
UNIVERSITÀ DI ROMA

Analysis, Design and **Implementation**
of a **Three-Layered...**

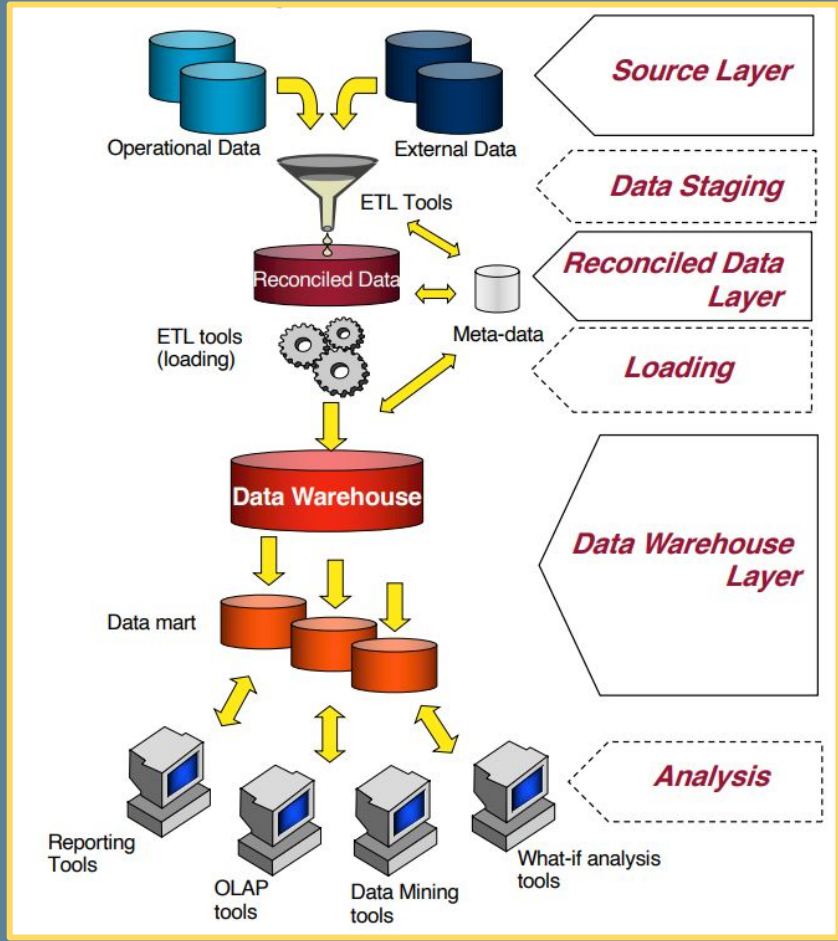
Videogame Data Warehouse

Longo Valerio 1655653
Papi Alessio 1761063

https://github.com/1655653/LSDM

Three-Layered DW Architecture

1. Data Sources Collection
2. Data Integration and Reconciliation
    a. Reconciled Schema
    b. ETL processing
3. Data Warehouse
    a. Conceptual Design
    b. Logical Design
    c. Physical Design
4. Analysis
    OLAP, Materialized Views
5. Qlik Sense data mart re-implementation



VGCHARTZ

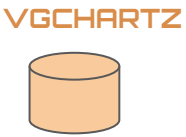
VGChartz delivers comprehensive game chart coverage, including:
sales data, news, reviews and game database for almost all game consoles.

From VGChartz we took:

- **vgsales-2021-04-09_16_51_31 (csv) :** list of the most sold 10000 games.
- **sales-by-console (csv)** : collection of **27** files, each holding vgsales information specific to every console supported by VGChartz.

Id	Game	Console	Publisher	Developer	Sales_Total	Na_sales	Pal_sales	Japan_sales	Other_Sales	Year	Last_Update

Id	Game	Console	Publisher	Developer	Sales	Year



GAMEDEV

Contains geographic-related information about videogame developing and publishing companies.

This dataset will be a crucial utility in the design of geographic and market related tasks.

From gamedev.net we took:

- **gamedev (csv)**

X_Coord	Y_Coord	Name	Type	City	Country	Website

VGCHARTZ GAMEDEV

METACRITIC

Metacritic is a website that aggregates reviews and averaged scores for films, TV shows, music albums, video games and formerly, books. We took from this source:

From metacritic we took:

- **metacritic (json) :** A list of all 18000 games present in the catalog

This dataset stresses out the averaged user score **(us)** and the metacritic score **(ms)** for each game of the dataset.

Id	Title	User_Score	MC_Score	Platform	Year

- **datagenreX (json)** : 16 different json files (based on 16 different genres)

Id	Title	Genre

KAGGLE

Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment.

From kaggle we took:

- **HistoricalEsportData (csv)** : historical information about 500 games and related earnings in tournaments.

Date	Game	Earnings	Players	#Tournaments

- **GeneralEsportData (csv)** : general information about 500 games and related earnings in tournaments.

Game	ReleaseDate	Genre	TotalEarnings	OnlineEarnings	PricedPlayers	TotalTournaments

CORGIS

CORGIS is a “collection of Real-time, Giant, Interesting, Situated Datasets”.

From corgis we took:

- **sales_playtime_video_games (csv) :** list of 1200 games

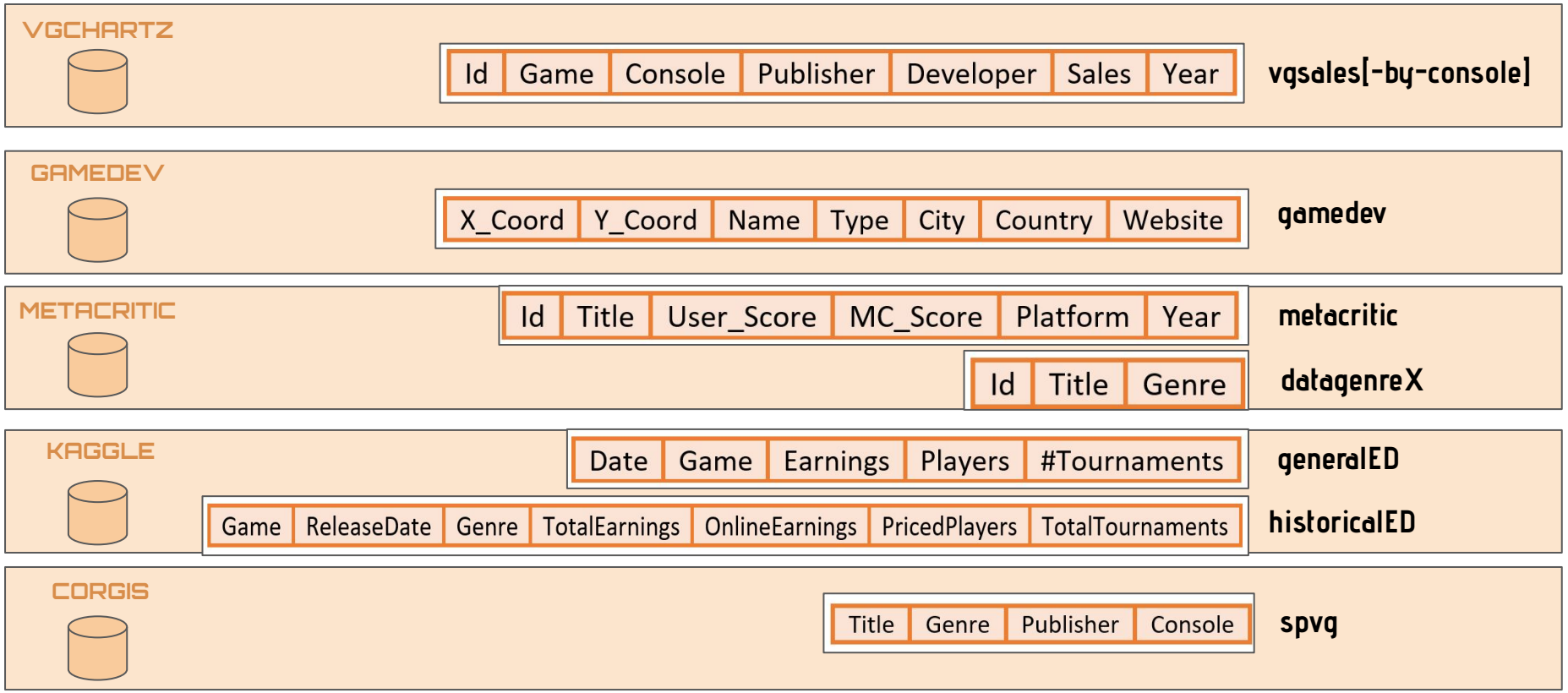
Title	Multiplayer?	Online?	Sequel?	Genre	Publisher	Price	Console	Campaign_Avg_Duration	Other_Metadata

↓

Title	Genre	Publisher	Console

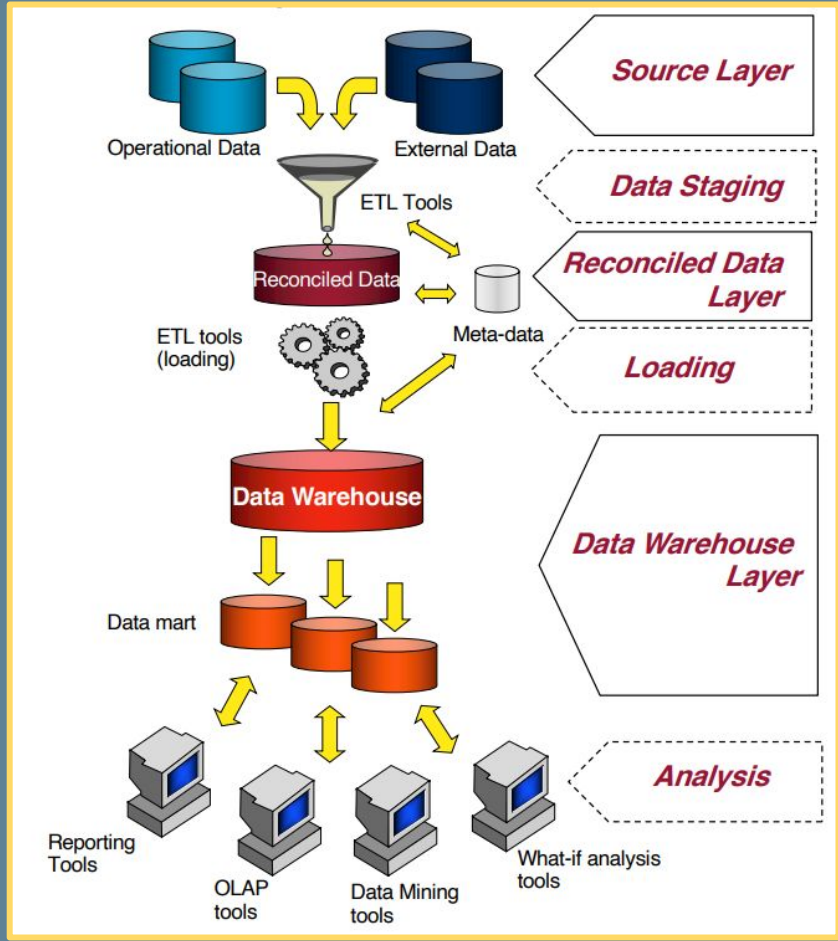
VGCHARTZ GAMEDEV METACRITIC KAGGLE CORGIS

Source Schema

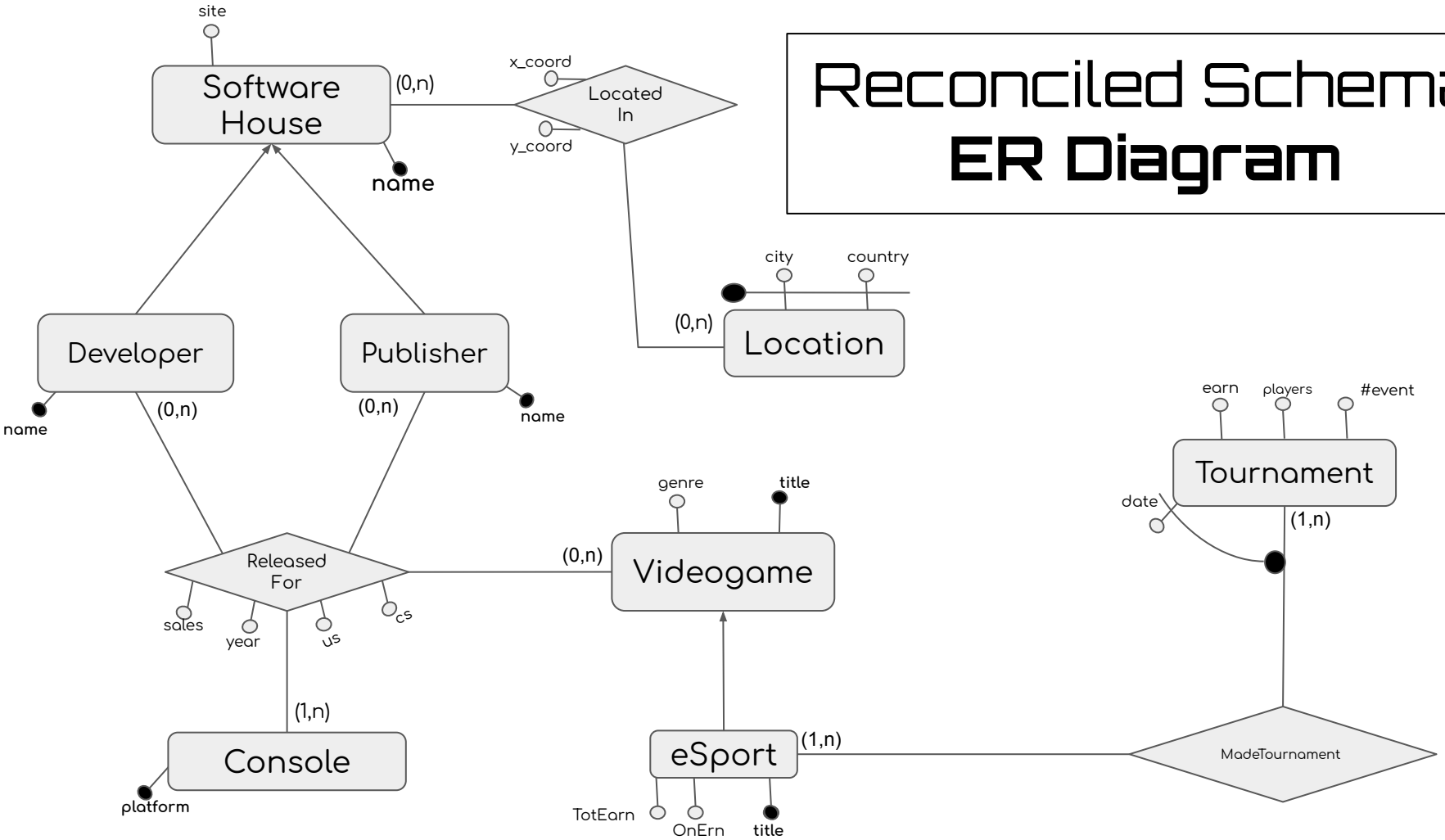


Three-Layered DW Architecture

1. Data Sources Collection
2. Data Integration and Reconciliation
   a. Reconciled Schema
   b. ETL processing
3. Data Warehouse
   a. Conceptual Design
   b. Logical Design
   c. Physical Design
4. Analysis
   OLAP, Materialized Views
5. Qlik Sense data mart re-implementation



Reconciled Schema
ER Diagram

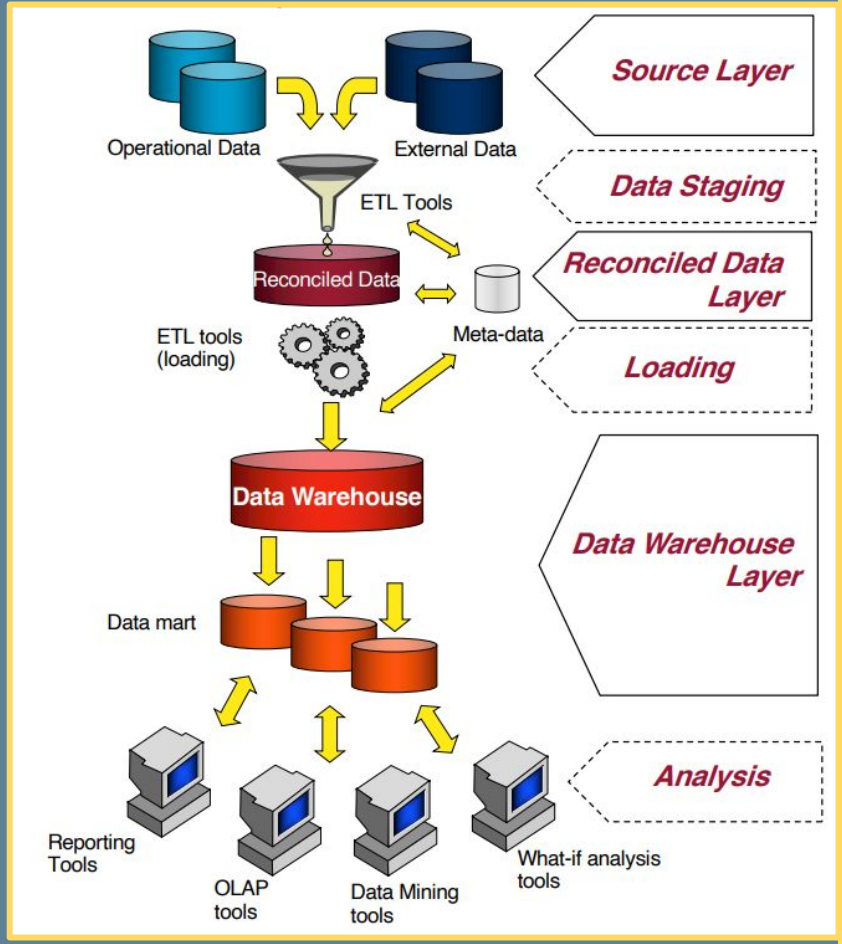


Reconciled Schema : **Relational Model**

- **SoftwareHouse**(name,site)
- **Location**(city, country)
- **LocatedIn**(softwarehouse, city, country, x_coord, y_coord)
  - FK: LocatedIn(softwarehouse) → SoftwareHouse(name)
  - FK: LocatedIn(city,country) → Location(city,country)
- **Videogame**(title, genre)
- **Console**(platform)
- **ReleasedFor**(game, console, us, cs, sales, year, dev, pub)
  - FK: ReleasedFor(game) → Videogame(title)
  - FK: ReleasedFor(console) → Console(platform)
  - FK: ReleasedFor(dev) → Developer(dev)
  - FK: ReleasedFor(pub) → Publisher(pub)
- **Tournament**(date, game, earnings, pricedplayers, num_events)
  - FK: Tournament(game) → eSport(title)
- **Developer**(name)
  - Developer(name) → SoftwareHouse(name)
- **Publisher**(name)
  - Publisher(name) → SoftwareHouse(name)
- **eSport**(title,TotalEarnings,OnlineEarnings)
  - eSport(title) → Videogame(title)

Three-Layered DW Architecture

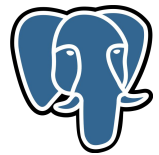
1. Data Sources Collection
2. Data Integration and Reconciliation
   a. Reconciled Schema
   b. ETL processing
3. Data Warehouse
   a. Conceptual Design
   b. Logical Design
   c. Physical Design
4. Analysis
   OLAP, Materialized Views
5. Qlik Sense data mart re-implementation

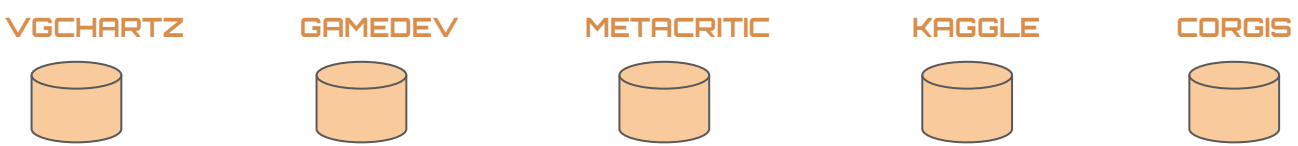


Data Exchange

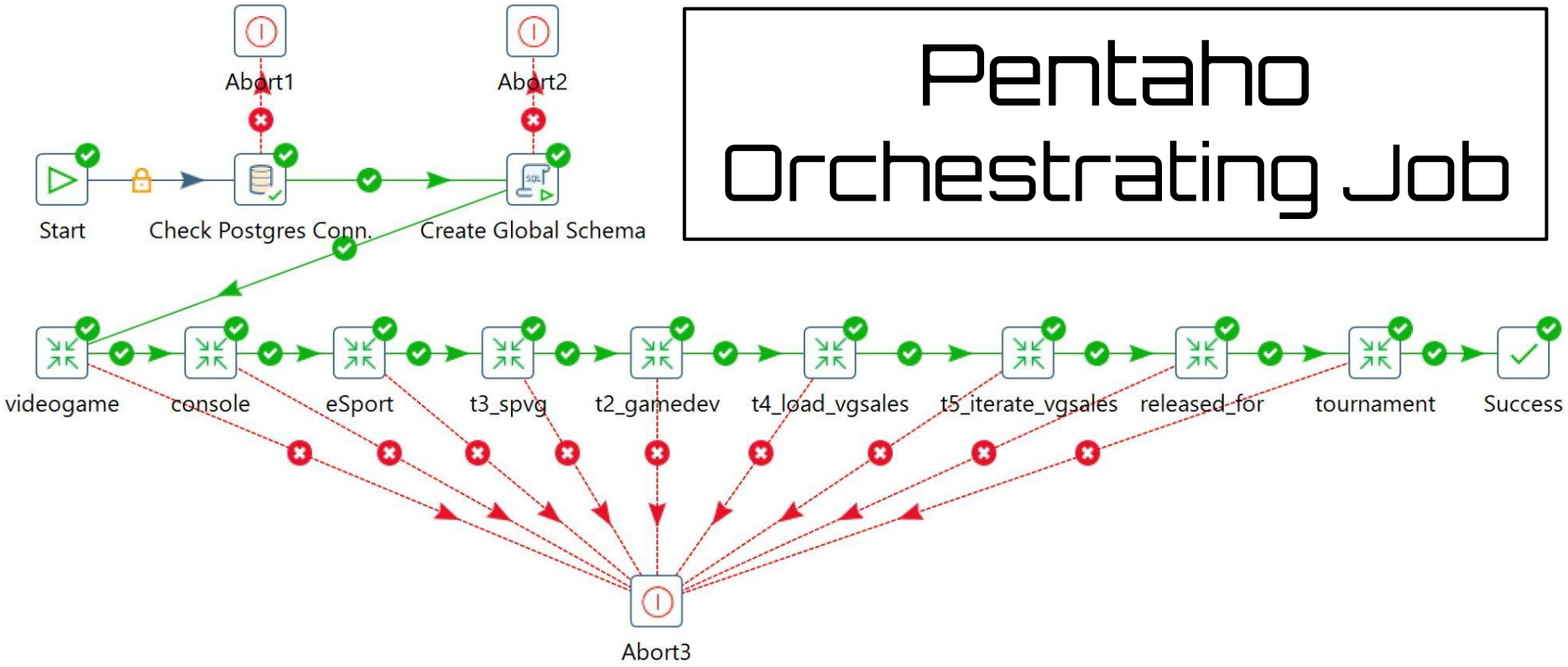
Materialization approach:

- ETL: Pentaho
  - Winning **Physical Heterogeneity** (wrappers + extraction)
  - **Cleansing**
  - Winning **Conceptual Heterogeneity** (transformation)
  - **PostgreSQL** Materialization (loading)

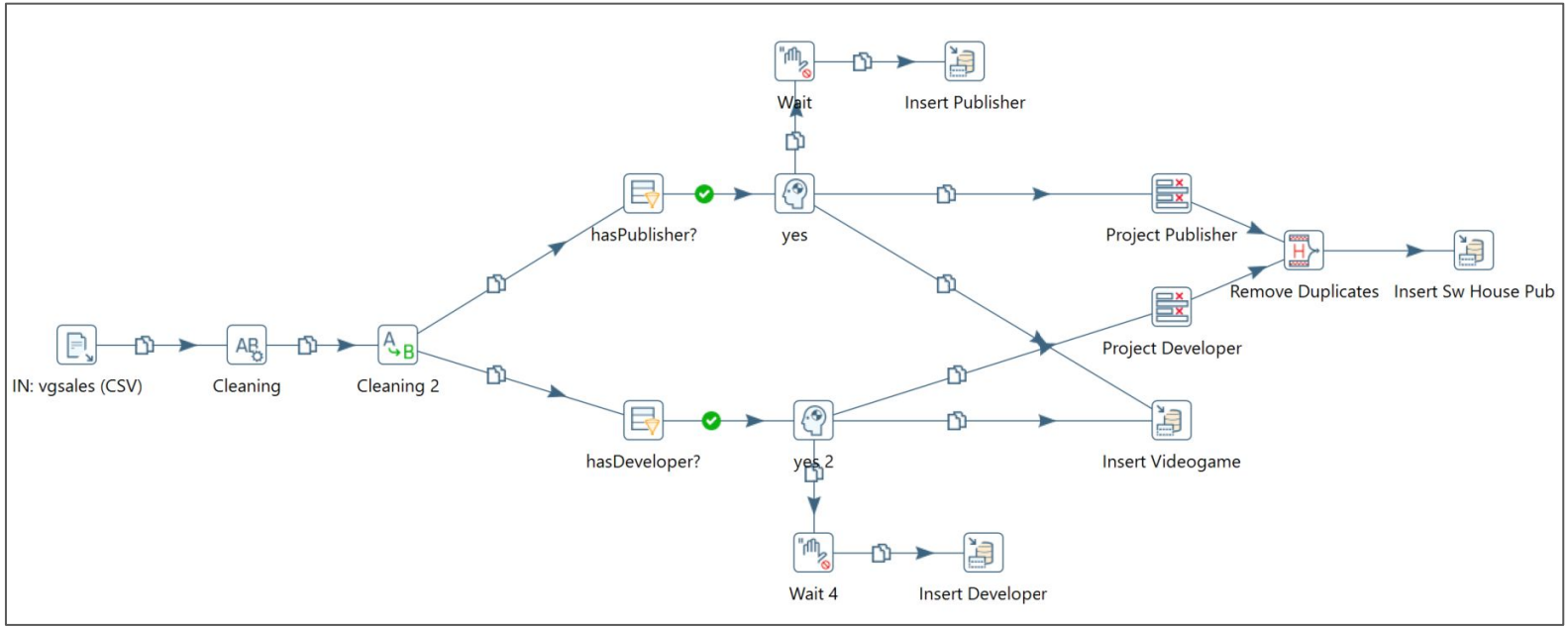




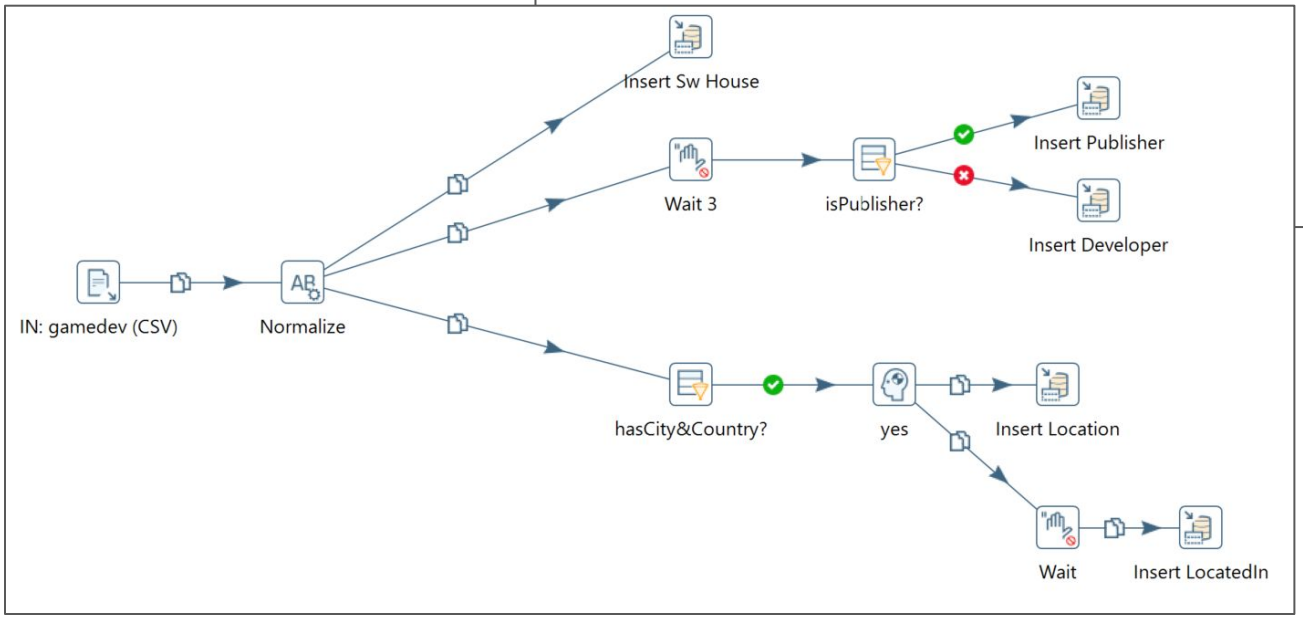
Pentaho Orchestrating Job

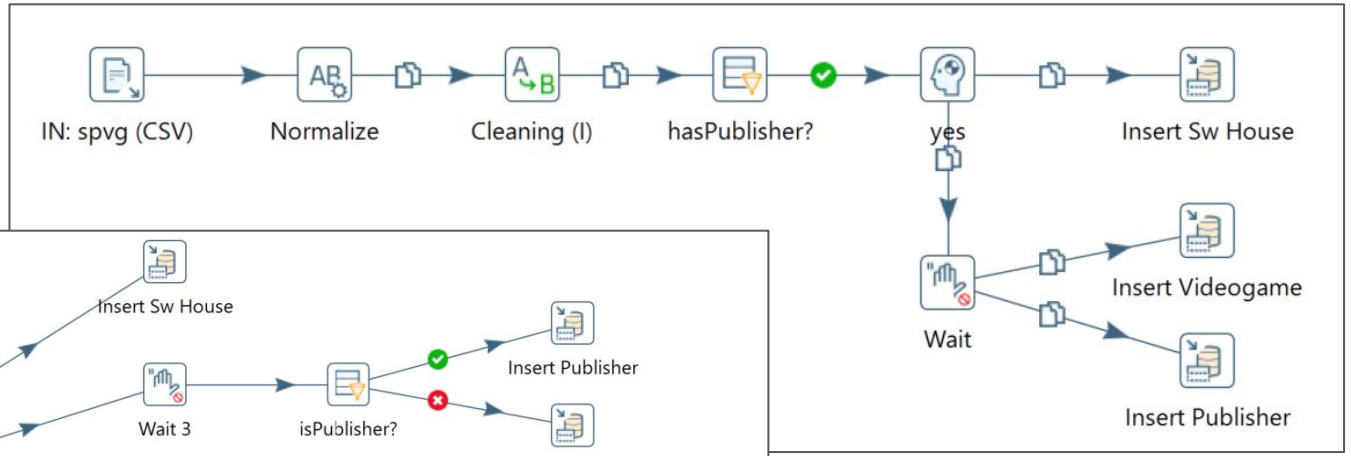


Pentaho: VGChartz Transformation

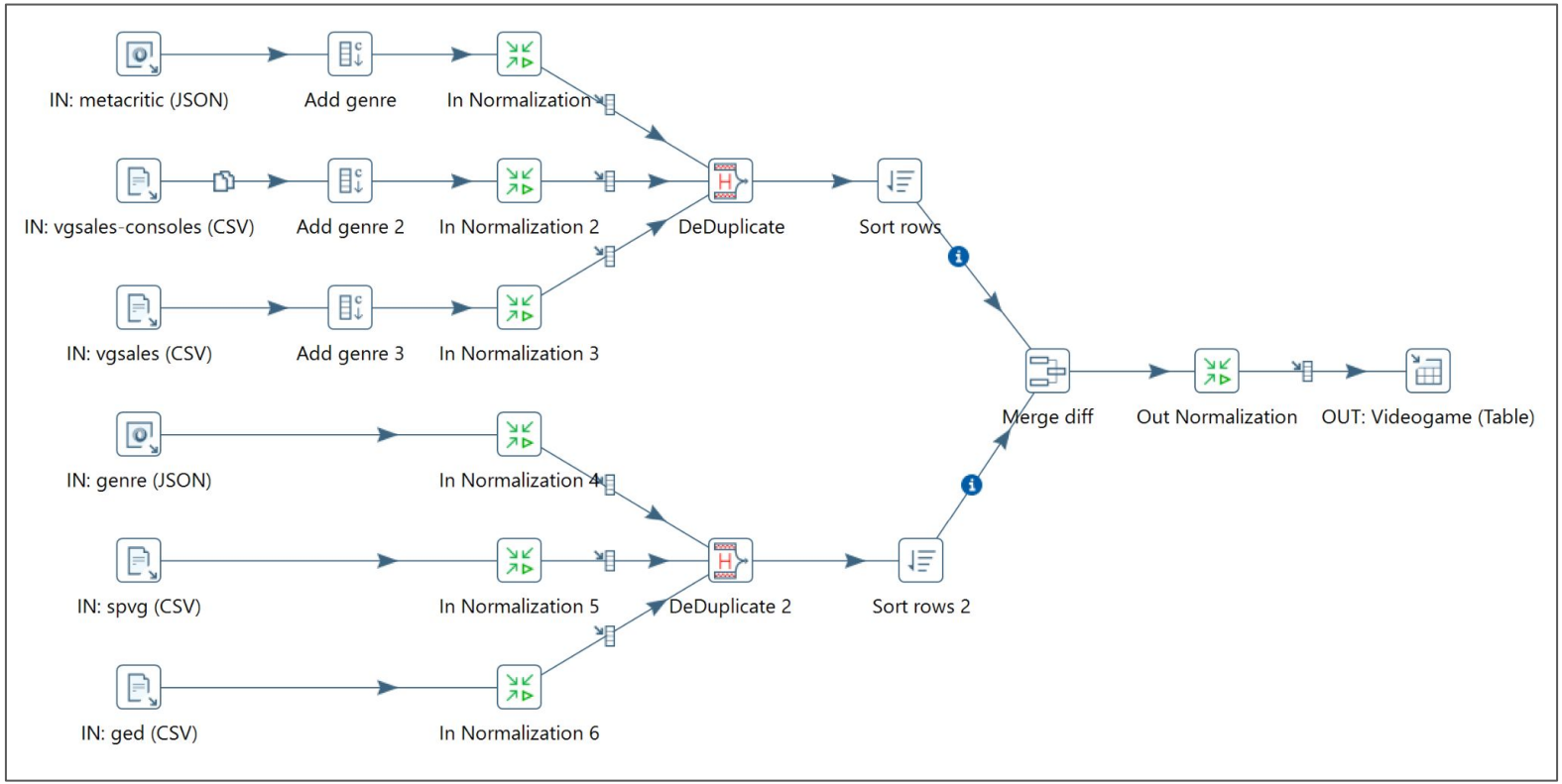


Pentaho: Gamedev, SPVG

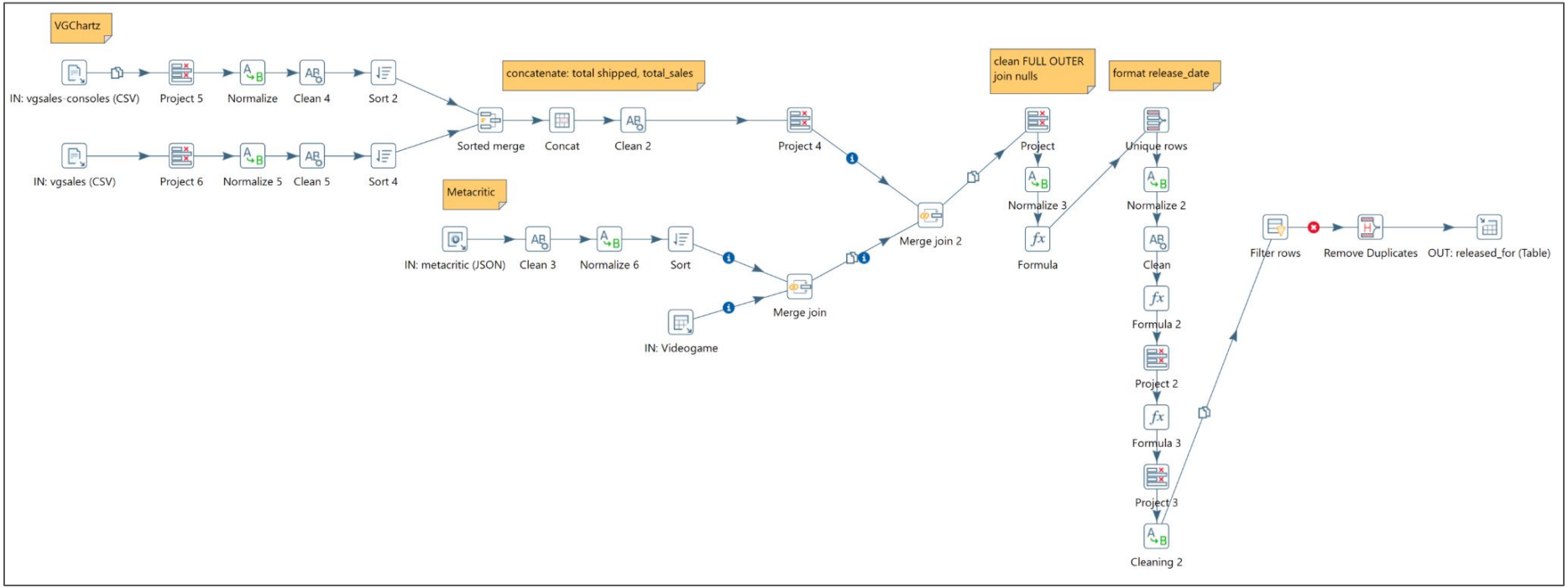




Pentaho: Videogame



Pentaho: ReleasedFor



PostgreSQL: Materialized Solution

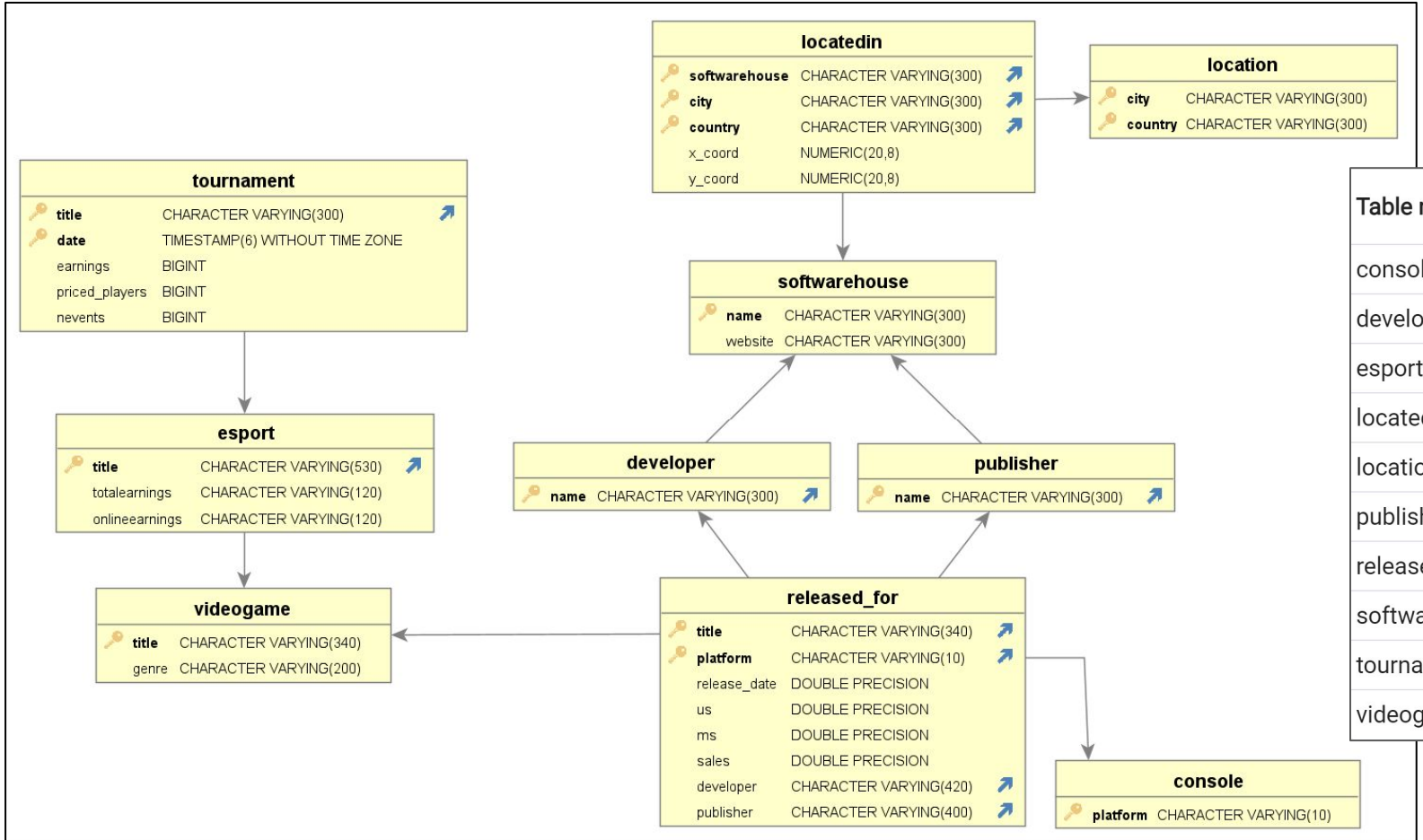
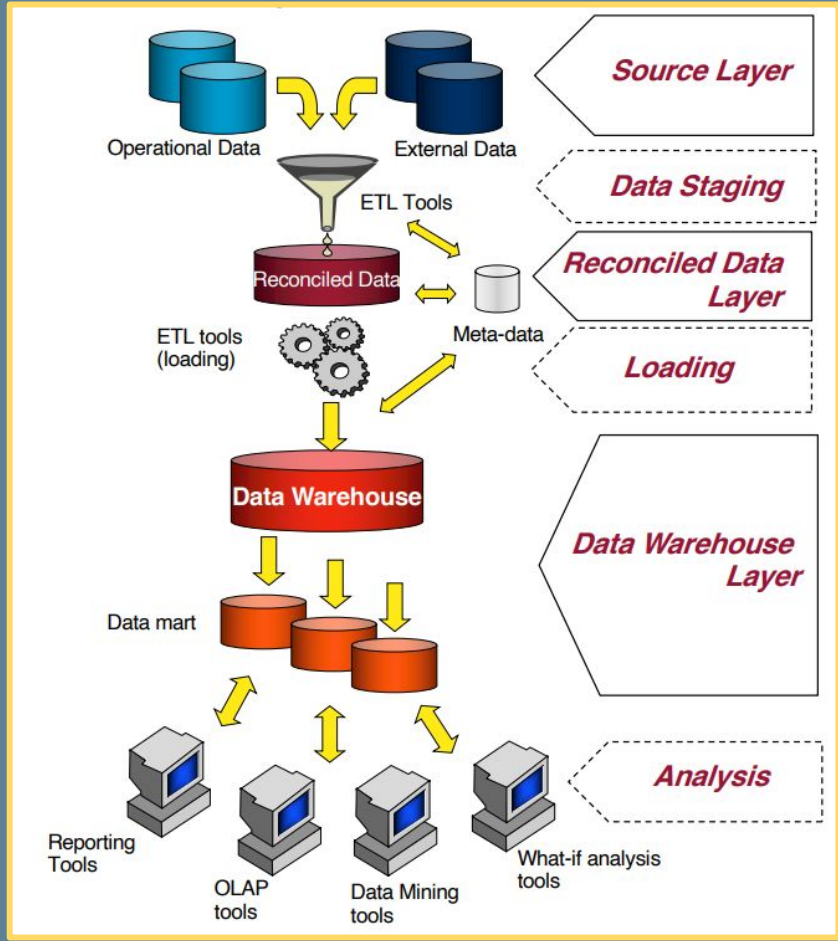


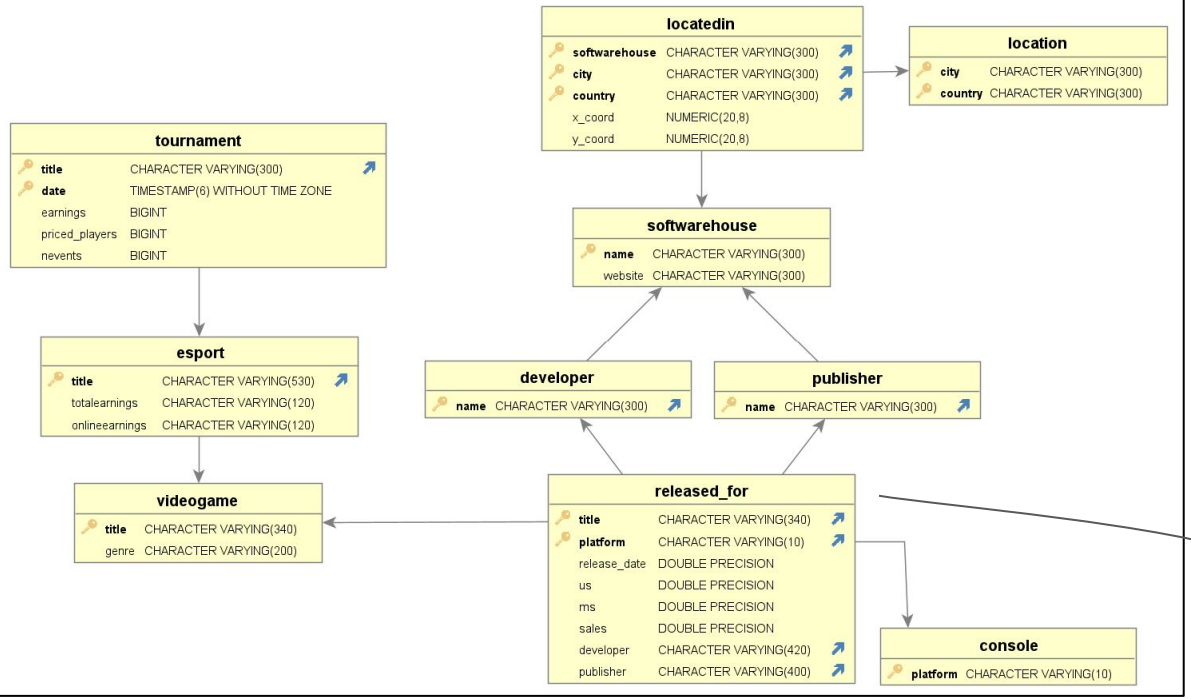
Table name	Tuples inserted
console	39
developer	8,050
esport	513
locatedin	4,140
location	1,052
publisher	1,852
released_for	37,535
softwarehouse	8,491
tournament	484
videogame	34,591

Three-Layered DW Architecture

1. Data Sources Collection
2. Data Integration and Reconciliation
   a. Reconciled Schema
   b. ETL processing
3. **Data Warehouse**
   a. **Conceptual Design (DFM)**
   b. **Logical Design (ROLAP)**
   c. **Physical Design (PostgreSQL)**
4. Analysis
   OLAP, Materialized Views
5. Qlik Sense data mart re-implementation

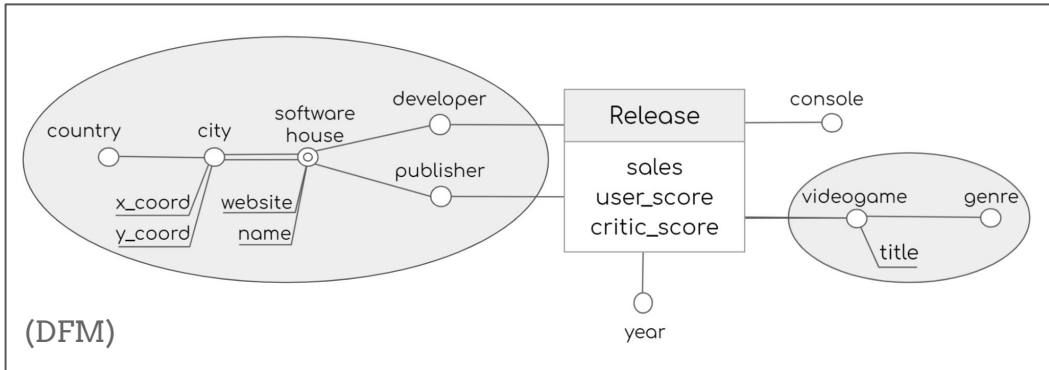


Conceptual Design (I)

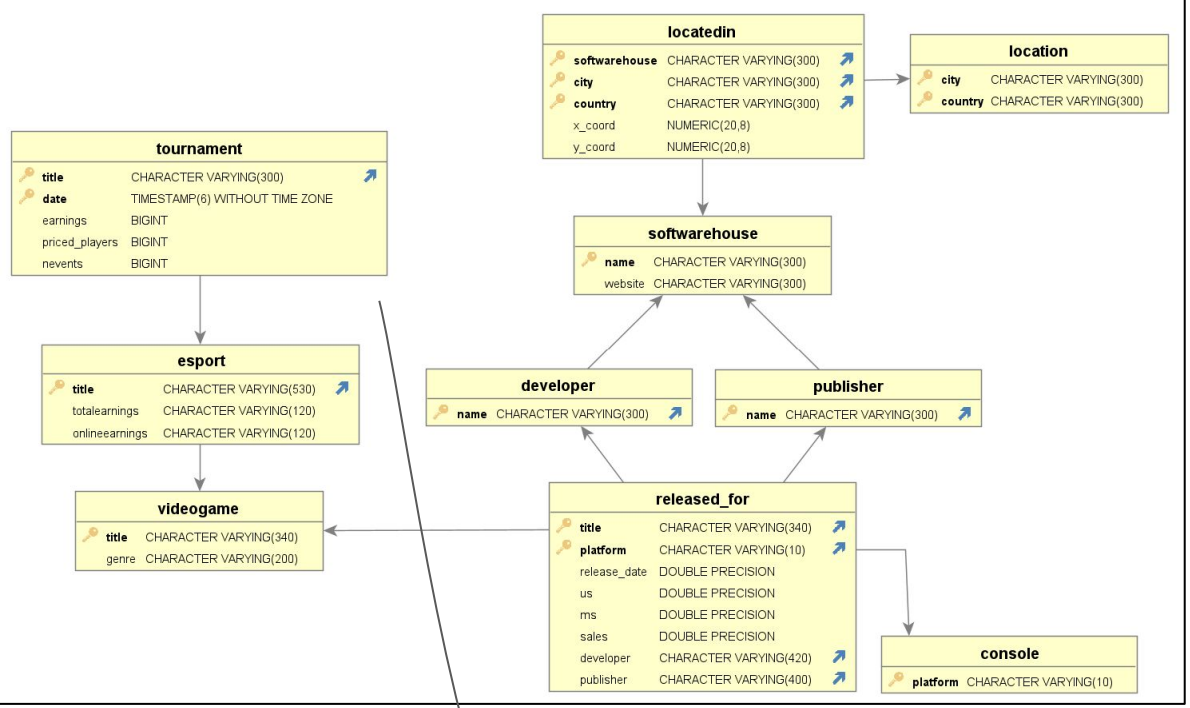


Reconciled Schema

Release Fact Schemata

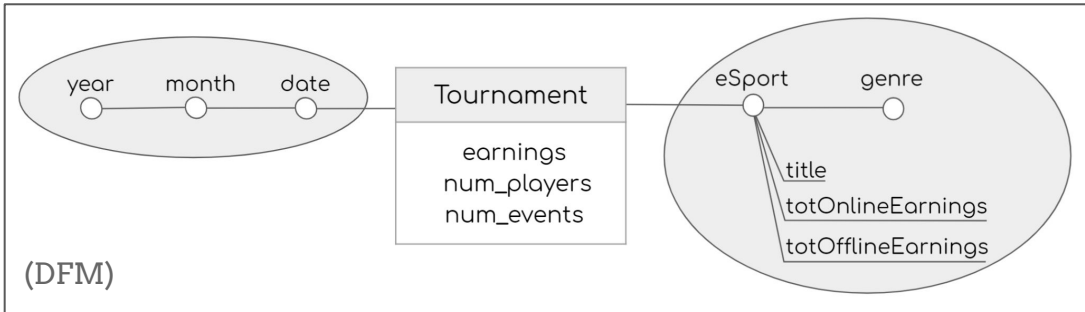


Conceptual Design (II)

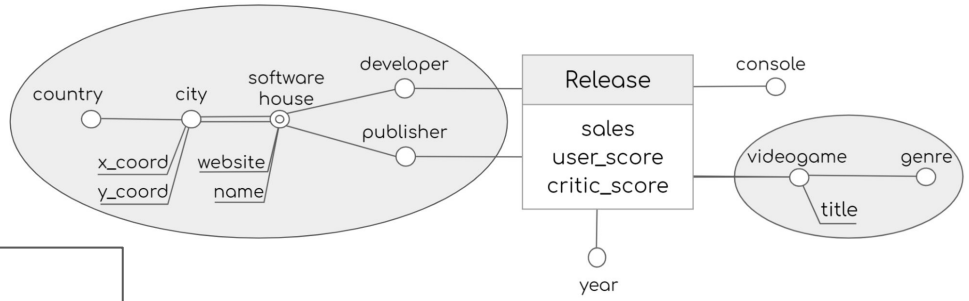


Reconciled Schema

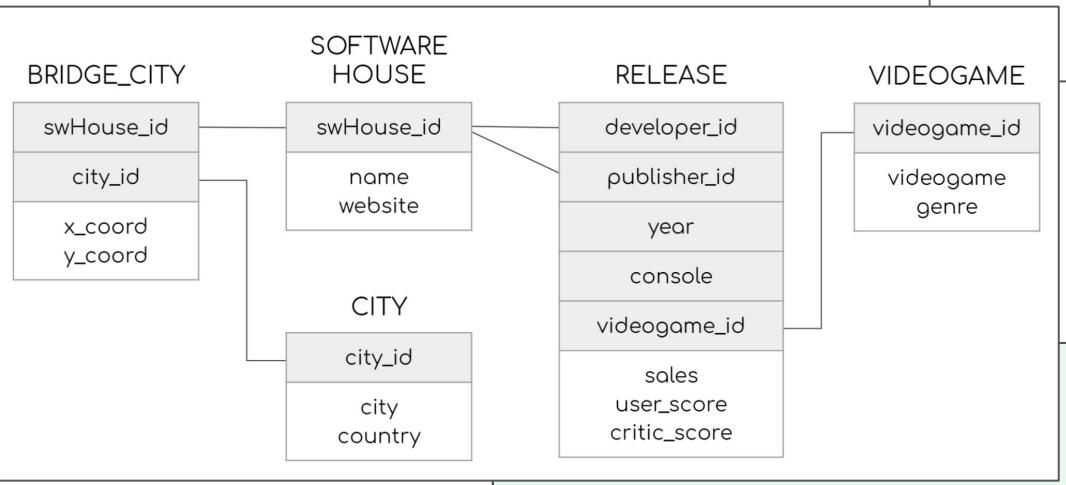
Tournament Fact Schemata



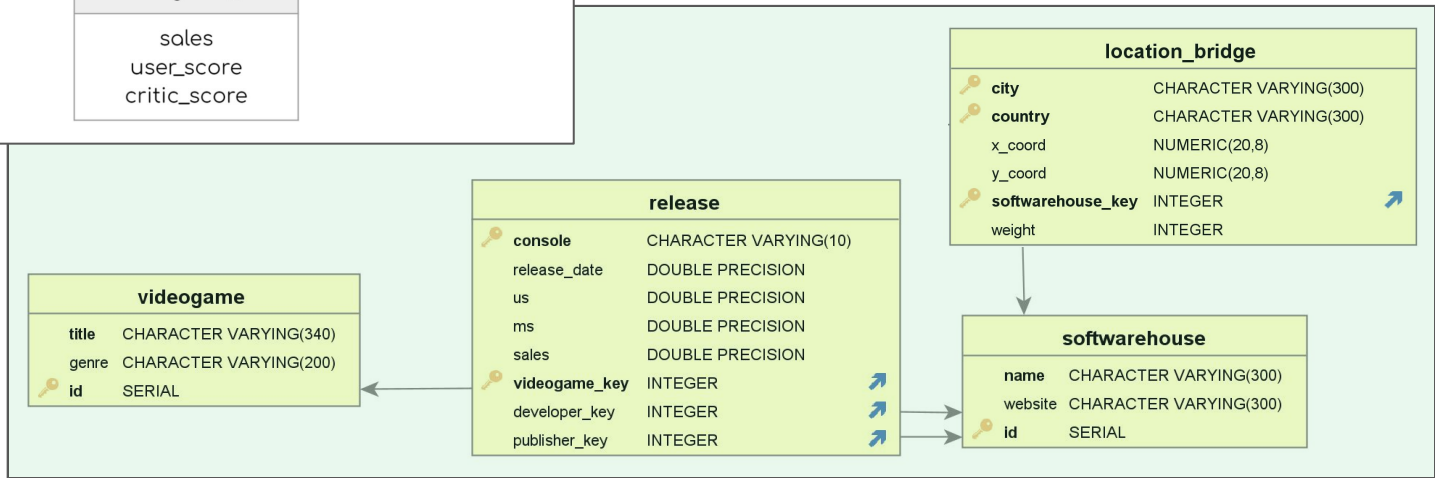
Conceptual Model:
Release Fact Schemata



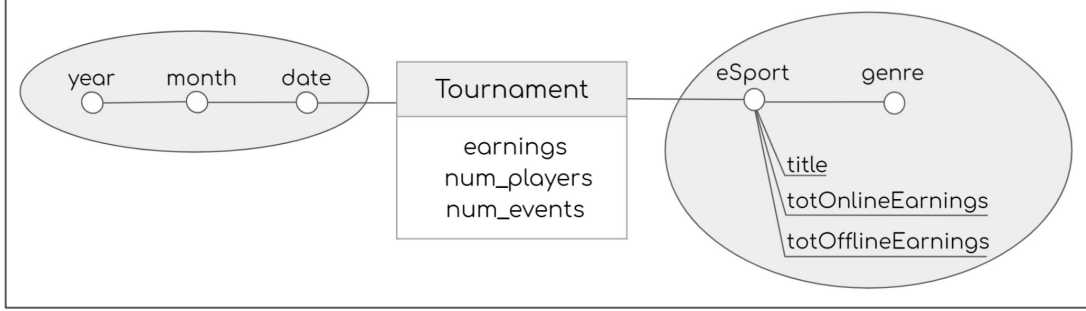
Logical Model:
Star Schema



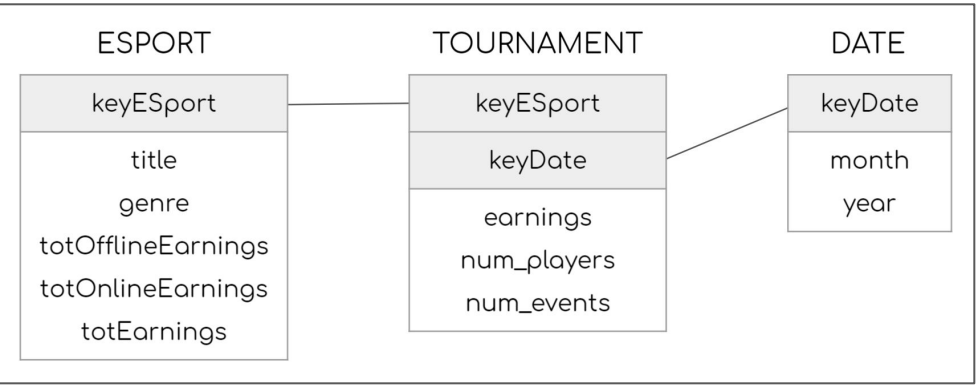
Physical Model:
Release Data Mart



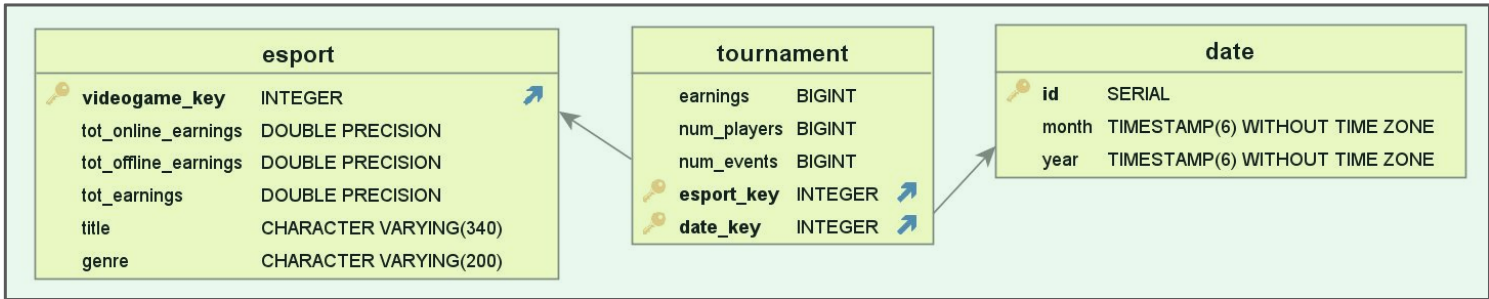
Conceptual Model:
Tournament Fact Schemata

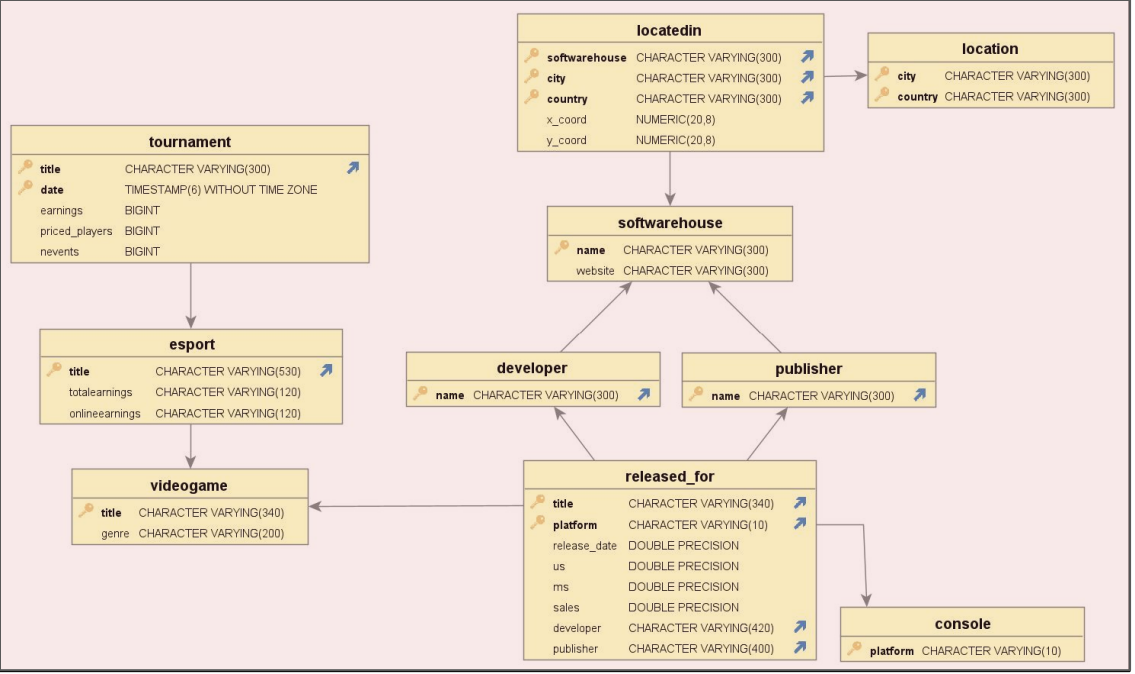


Logical Model:
Star Schema



Physical Model:
Tournament Data Mart

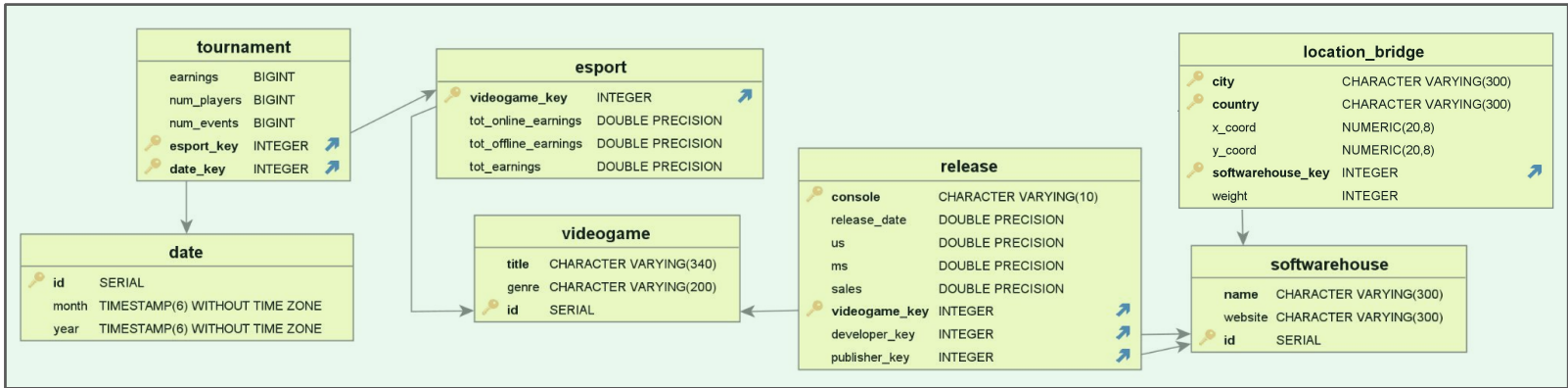




Reconciled Database

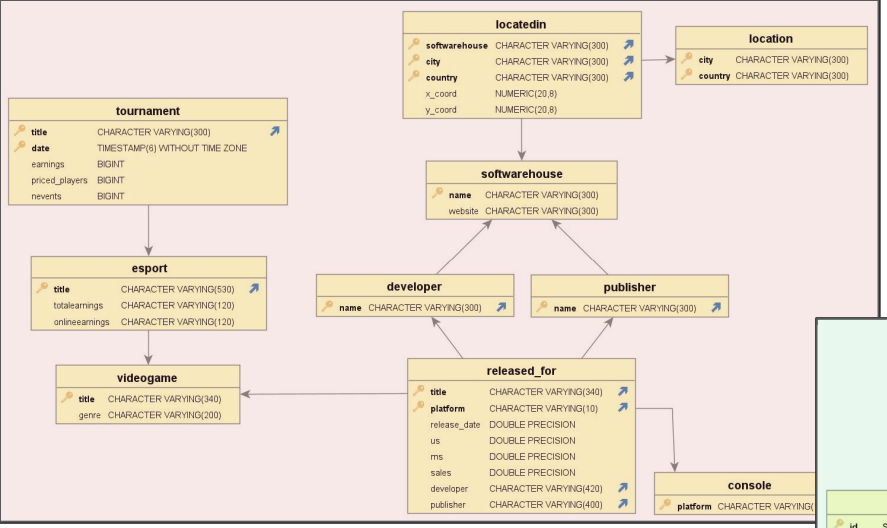
- Tables **removal**
- **Surrogate keys**
- Bridge table: **weight**
- Created: **Date** dimension table

Data Warehouse



Generation Workflow

gen_warehouse.bat



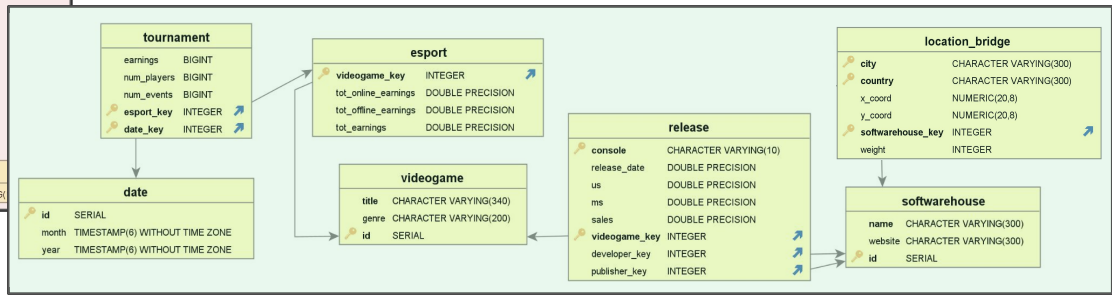
Reconciled Database

create_global_schema.sql

j1_main.kjb

reconciled_to_dw.sql

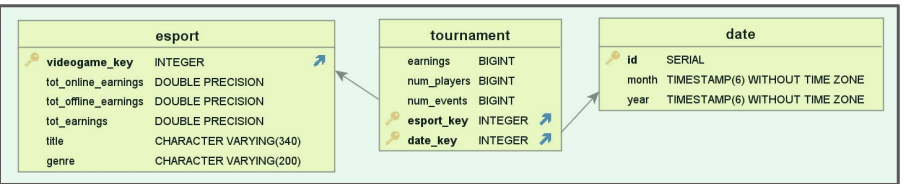
Data Warehouse



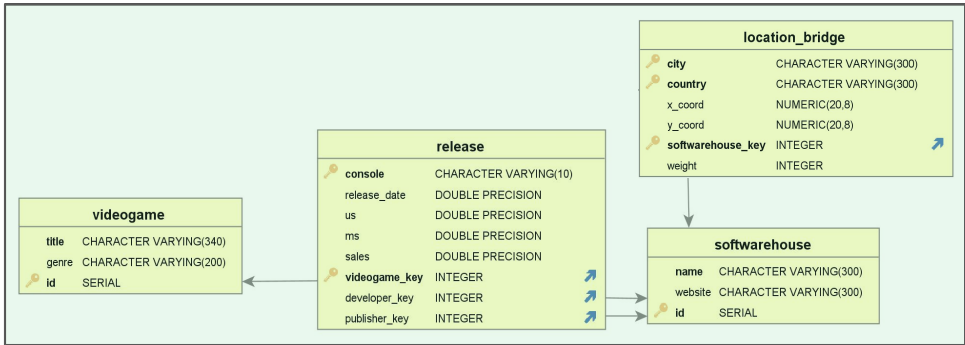
dw_to_tournament.sql

dw_to_release.sql

Tournament Data Mart



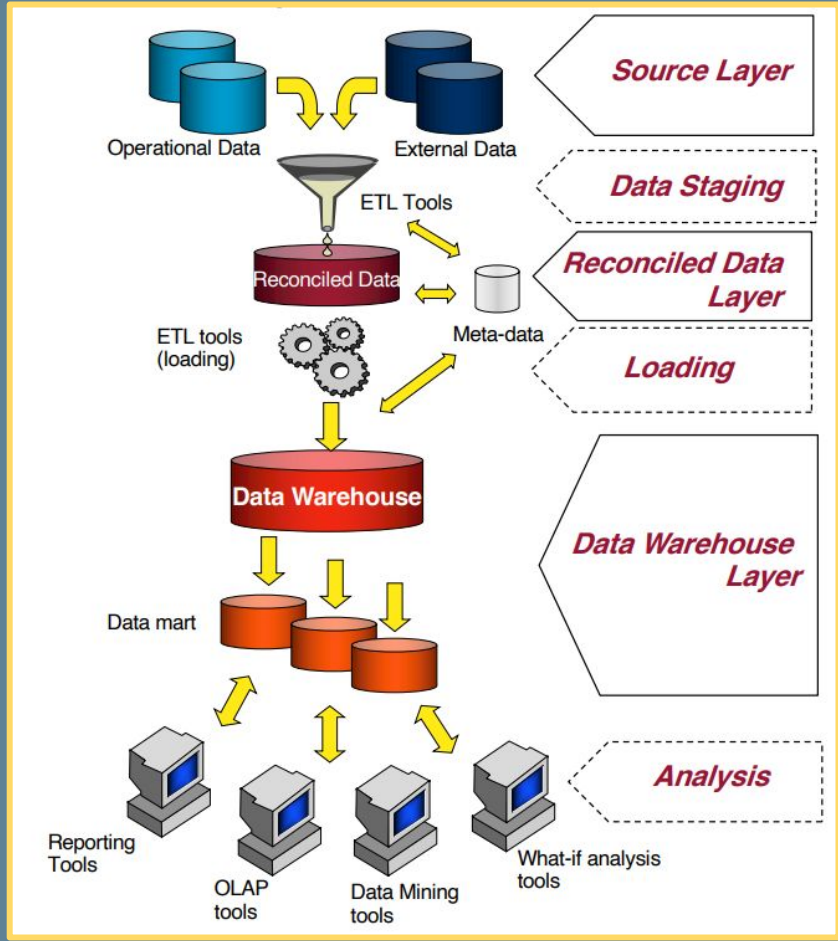
Release Data Mart



Three-Layered DW Architecture

1. Data Sources Collection
2. Data Integration and Reconciliation
   a. Reconciled Schema
   b. ETL processing
3. Data Warehouse
   a. Conceptual Design
   b. Logical Design
   c. Physical Design
4. **Analysis**

   OLAP, Materialized Views
5. Qlik Sense data mart re-implementation



Analysis

The main purpose of our DW is to provide an **OLAP interface** towards the previously designed **multidimensional model**.

Interesting **OLAP analysis**:

- **Release** multidimensional cube: roll up or slice/dice by *videogame genre* and/or *publishing/developing country*.
- **Release** multidimensional cube: slice/dice by *console* and/or *year*.
- **Tournament** multidimensional cube: roll up or slice/dice by *month* and/or *eSport genre*.

Interesting **data insights**:

1. Publishers who earned the most from release sales, per year?
2. Countries who earned the most from release sales, per year?
3. Countries who published most videogames?
4. Pair videogame genre - publisher with the highest ratings?
5. Which months are the most profitable for tournaments?

OLAP: Release (1)

Show the **RELEASE** multidimensional cube as is:

```
select *
from release_mart."release";
```

	videogame_key [PK] integer	developer_key integer	publisher_key integer	console [PK] characte	release_date double precision	sales double pre	us double	ms double p
1	4	6486	2	PS3	2012	0.34	42	41
2	4	6486	2	X360	2012	0.29	45	45
3	6	4257	2	PS3	2008	1.14	66	65
4	6	3915	2	Wii	2008	0.65	73	54
5	6	4257	2	X360	2008	1.48	70	65
6	7	937	3827	PS	2000	0.53	48	51
7	8	4283	3827	N64	2000	1.55	78	81
8	8	5525	3827	PS	2000	0.92	67	61
9	22	4083	123	DS	2010	0.67	65	70
10	64	3873	123	NS	2017	3.18	48	58
11	65	3330						
12	86	8305						

Successfully run. Total query runtime: 156 msec. 37535 rows affected.

OLAP: Release (2)

Roll up the **RELEASE** multidimensional cube by videogame genre and publishing country:

```
select genre, country, sum(sales), avg(us), avg(ms)
from release_mart.release as x
join release_mart.location_bridge as y on x.publisher_key = y.softwarehouse_key
join release_mart.videogame as z on x.videogame_key = z.id
group by genre, country;
```

	genre character varying (2	country character varyin	sales numeric (1	us numeric (1	ms numeric (10
1	real-time	singapore	26.15	92.00	90.00
2	action	netherlands	1.05	74.75	79.50
3			.00	74.00	93.00
4			.00	74.00	93.00
5			.55	70.25	63.50
6	real-time	spain	26.42	92.00	90.00
7	racing	united states	42.90	76.33	78.50
8	role-playing game	taiwan	12.00	74.00	93.00
9	role-playing	china	50.63	65.50	87.50
10	puzzle				

✔ Successfully run. Total query runtime: 141 msec. 349 rows affected.

OLAP: Release (3)

Roll up the **RELEASE** multidimensional cube by videogame genre and **dice** by year and console:

```
select genre, x.console, x.release_date, sum(sales), avg(us), avg(ms)
from release_mart.release as x
join release_mart.videogame as z on x.videogame_key = z.id
where (x.console = 'X360' or x.console = 'PS3' or x.console = 'PC') and
    2020 >= x.release_date and x.release_date >= 2000
group by genre, x.console, x.release_date;
```

	genre character va	console character vary	release_date double precision	sales numeric (1	us numeric (	ms numeric
1	action	PC	2011	56.79	71.50	74.25
2	action	PS3	2011	44.82	61.67	60.43
3			2010	43.47	72.83	70.33
4			2012	41.05	68.13	71.13
5			2011	34.49	64.25	60.78
6			2010	34.03	70.80	67.80
7			2012	33.26	62.89	68.22
8	role-playi...	PC	2012	25.06	74.50	79.75
9	action	PC	2013	24.72	65.75	70.13
10	sports	PS3	2010	23.50	70.67	74.33
11	sports	X360	2010	22.44	72.40	66.00
12	sports	PS3	2012	19.30	62.50	71.00
13	action	PC	2014	19.04	69.86	75.00
14	a					

✔ Successfully run. Total query runtime: 123 msec. 533 rows affected.

OLAP: Tournament (4)

Show the **TOURNAMENT** multidimensional cube as is:

```
select *
from tournament_mart.tournament;
```

	date_key [PK] integer	esport_key [PK] integer	earnings bigint	num_players bigint	num_events bigint
1	1	493	210	3	1
2	2	737	2000	1	1
3	3	739	35336	2	1
4	4	744	20000	3	1
5	5	749	100	3	1
6	6	751	49000	8	2
7	7	901	20	3	1
8	8	1030	40000	1	1
9	9	1215	36000	8	3
10	10	1453	212829	116	8
11	11	1576	210	3	1
12	11	1491	160	3	1
13	12	1575	1000	4	1
14	14	1577	250	3	1
15	1	3668	50000	16	1
16	1	30690	190	3	1
17					
18					

Successfully run. Total query runtime: 78 msec. 484 rows affected.

OLAP: Tournament (5)

Roll up the **TOURNAMENT** multidimensional cube eSport genre and year and **dice** by year:

```
select genre, year, sum(earnings), sum(num_players), sum(num_events)
from tournament_mart.tournament as x
join tournament_mart.esport as y on x.esport_key = y.videogame_key
join tournament_mart.date as z on x.date_key = z.id
where 2018 >= year and year >= 2015
group by genre, year;
```

	genre character varying (200)	year text	tot earnings numeric	priced players numeric	events numeric
1	strategy	2018	220	4	1
2	first-person shooter	2016	422311	110	15
3	racing	2017	20163	12	1
				243	2
				6	12
				152	11
				3	1
				82	21
				121	13
10	racing	2016	4617	32	7
11	battle royale	2015	173328	20	2
12	sports	2017	31462	17	4
13	fighting game				
14	collectible card game				

Successfully run. Total query runtime: 88 msec. 35 rows affected.

Insight (1)

PUBLISHERS WHO EARNED THE MOST BY **SALES, PER YEAR** ?

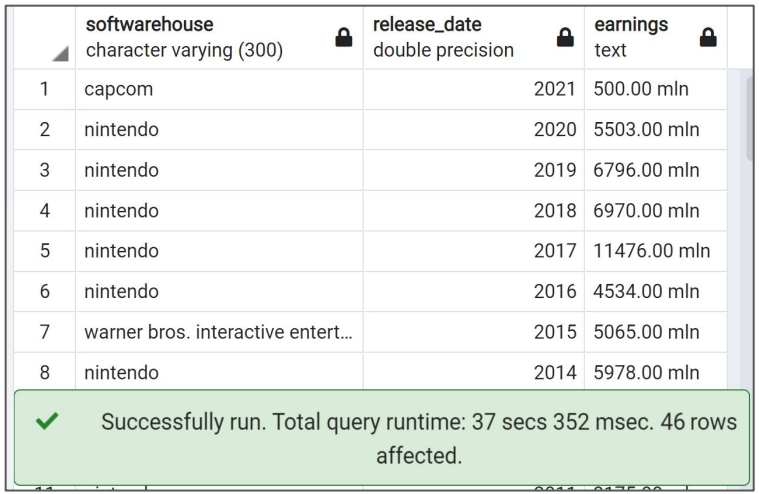
```
select x.publisher_key, x.release_date, sum(sales)
from release_mart.release as x
where x.sales notnull and x.release_date notnull
group by x.publisher_key, x.release_date
having sum(sales) >= (
    select sum(sales)
    from release_mart.release as y
    where x.release_date = y.release_date and y.sales notnull
    group by y.publisher_key
    order by sum(sales) desc limit 1
)
order by x.release_date desc
```

	softwarehouse character varying (300)	release_date double precision	earnings text
1	capcom	2021	500.00 mln
2	nintendo	2020	5503.00 mln
3	nintendo	2019	6796.00 mln
4	nintendo	2018	6970.00 mln
5	nintendo	2017	11476.00 mln
6	nintendo	2016	4534.00 mln
7	warner bros. interactive entert...	2015	5065.00 mln
8	nintendo	2014	5978.00 mln

✔ Successfully run. Total query runtime: 19 secs 512 msec. 46 rows affected.

Optimization (1)

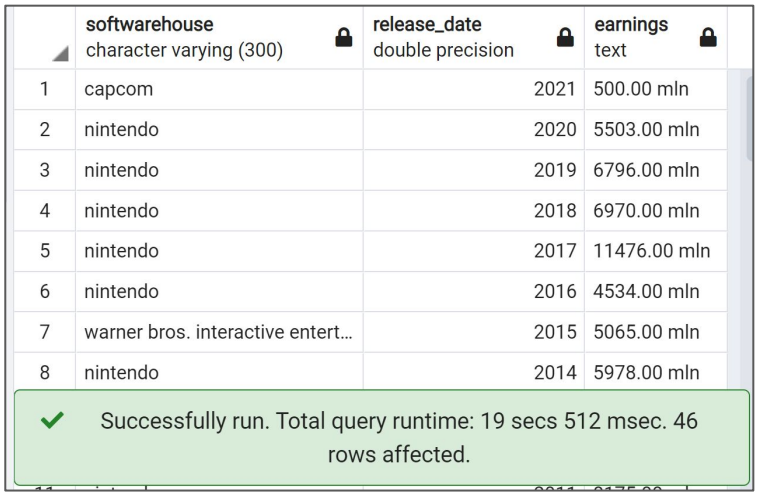
PUBLISHERS WHO EARNED THE MOST BY **SALES, PER YEAR** ?



	softwarehouse character varying (300)	release_date double precision	earnings text
1	capcom	2021	500.00 mln
2	nintendo	2020	5503.00 mln
3	nintendo	2019	6796.00 mln
4	nintendo	2018	6970.00 mln
5	nintendo	2017	11476.00 mln
6	nintendo	2016	4534.00 mln
7	warner bros. interactive entert...	2015	5065.00 mln
8	nintendo	2014	5978.00 mln

Successfully run. Total query runtime: 37 secs 352 msec. 46 rows affected.

On the Reconciled Schema.



	softwarehouse character varying (300)	release_date double precision	earnings text
1	capcom	2021	500.00 mln
2	nintendo	2020	5503.00 mln
3	nintendo	2019	6796.00 mln
4	nintendo	2018	6970.00 mln
5	nintendo	2017	11476.00 mln
6	nintendo	2016	4534.00 mln
7	warner bros. interactive entert...	2015	5065.00 mln
8	nintendo	2014	5978.00 mln

Successfully run. Total query runtime: 19 secs 512 msec. 46 rows affected.

On the Release Data Mart.

Insight (2)

COUNTRIES WHO EARNED THE MOST FROM **SALES PER YEAR** ?

```sql
select release_date, country, concat(ceil(sum(sales/weight)*100),' mln')
from (
    select country, release_date, sales, weight
    from release_mart.release as x
    join release_mart.location_bridge as y on x.publisher_key = y.softwarehouse_key) as x
group by release_date, country
having sum(sales/weight) >= (
    select sum(sales/weight)
    from (
        select country, release_date, sales, weight
        from release_mart.release as x
        join release_mart.location_bridge as y on x.publisher_key = y.softwarehouse_key) as y
    where y.release_date = x.release_date
    group by country
    order by sum(sales/weight) desc limit 1
) order by release_date desc
```

Weighted Query

release_date double precision	country character varying (300)	concat text
2021	japan	250 mln
2021	united states	250 mln
2020	japan	5531 mln
2019	japan	6815 mln
2018	japan	8152 mln
2017	japan	13474 mln
2016	japan	6746 mln
2015	japan	4553 mln
2014	japan	8355 mln
2013	japan	9832 mln
2012	japan	7997 mln

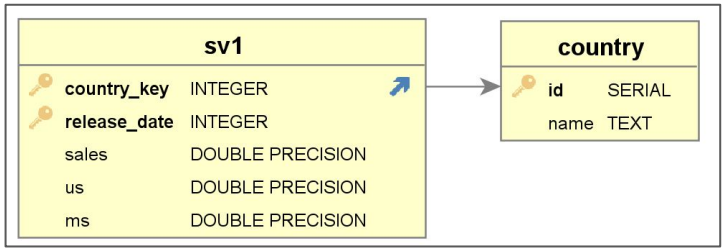
✔ Successfully run. Total query runtime: 3 secs 120 msec. 52 rows affected.

Optimization (2)

COUNTRIES WHO EARNED THE MOST FROM **SALES**, **PER YEAR** ?

We materialized a **Secondary View** to optimize the execution time of the second insight query:

- We performed a Roll Up operation on the Release multidimensional cube on the whole *Console, Videogame, Developer* dimensions, whereas *Publisher* → *Country* and *Year* remained as is..
- We materialized the Secondary View employing the **Constellation Schema**: a separate, aggregated fact table was created, as well as the new dimensional table for publishers, at the country granularity.

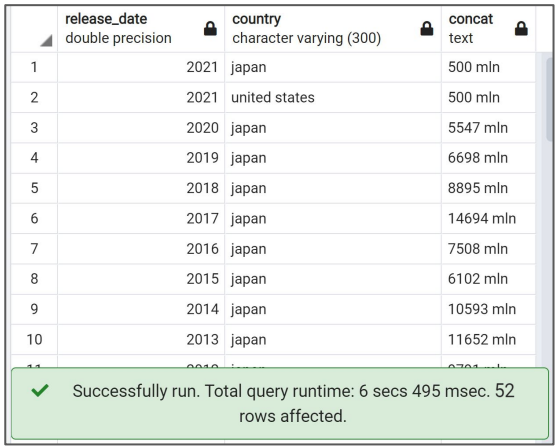


Constellation Schema

```
select release_date, country_key, concat(ceil(sales*100),' mln')
from release_mart.sv1 as x
group by country_key, x.release_date
having sum(sales) >= (
    select sum(sales)
    from release_mart.sv1 as y
    where y.release_date = x.release_date
    group by country_key
    order by sum(sales) desc limit 1
) order by release_date desc
```

Optimization (2)

COUNTRIES WHO EARNED THE MOST FROM **SALES PER YEAR** ?

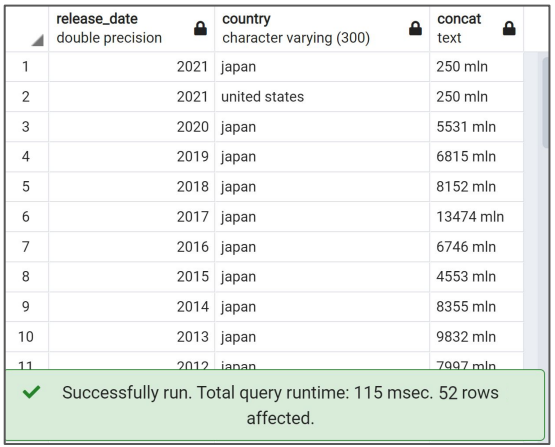


	release_date double precision	country character varying (300)	concat text
1	2021	japan	500 mln
2	2021	united states	500 mln
3	2020	japan	5547 mln
4	2019	japan	6698 mln
5	2018	japan	8895 mln
6	2017	japan	14694 mln
7	2016	japan	7508 mln
8	2015	japan	6102 mln
9	2014	japan	10593 mln
10	2013	japan	11652 mln

On the **Reconciled Schema**.

	release_date double precision	country character varying (300)	concat text
1	2021	japan	250 mln
2	2021	united states	250 mln
3	2020	japan	5531 mln
4	2019	japan	6815 mln
5	2018	japan	8152 mln
6	2017	japan	13474 mln
7	2016	japan	6746 mln
8	2015	japan	4553 mln
9	2014	japan	8355 mln
10	2013	japan	9832 mln

On the **Release Data Mart**.



	release_date double precision	country character varying (300)	concat text
1	2021	japan	250 mln
2	2021	united states	250 mln
3	2020	japan	5531 mln
4	2019	japan	6815 mln
5	2018	japan	8152 mln
6	2017	japan	13474 mln
7	2016	japan	6746 mln
8	2015	japan	4553 mln
9	2014	japan	8355 mln
10	2013	japan	9832 mln

On the **Secondary View** (constellation schema).

Insight (3)

COUNTRIES WHO PUBLISHED MOST VIDEOGAMES ?

impact query: resulting quantities are higher because the weight could not be used

```
select country , count(*)
from(
    select country, x.videogame_key
    from release_mart.release as x
    join release_mart.location_bridge as y on x.publisher_key = y.softwarehouse_key
    where sales notnull and release_date notnull
) as u
group by country
order by count(*) desc
```

	country character varying (300)	count bigint
1	japan	3047
2	united states	2891
3	germany	1643
	england	537
	china	134
	south korea	134
	singapore	131
	france	126
	canada	107
	poland	103
	netherlands	68
	italy	58
13	spain	54
14	austria	50
15	australia	36
16	sweden	35
17	brazil	32
18	taiwan	31
19	ireland	30
20	peru	26
21	switzerland	15

✔ Successfully run. Total query runtime: 53 msec. 48 rows affected.

Insight (4)

PAIR **GENRE-PUBLISHER** WITH THE HIGHEST RATINGS

```sql
select genre, "name",
round(avg(us)::numeric,2) as avg_us,
round(avg(ms)::numeric,2) as avg_ms
from release_mart."release" as x
join release_mart.videogame ON videogame.id = x.videogame_key
join release_mart.softwarehouse ON softwarehouse.id = x.publisher_key
where us notnull and ms notnull and genre notnull
group by genre,"name"
order by avg(us) DESC
```

	genre character varying (	name character varying (300)	avg_us numeric	avg_ms numeric
	role-playing	square	92.00	84.00
	real-time	blizzard entertainment	92.00	90.00
	real-time	microsoft	91.00	92.00
	real-time	gt interactive	88.00	86.00
	role-playing	working designs	88.00	86.00
	strategy	microsoft	87.00	83.00
	simulation	agetec	85.50	69.33
	racing	bam! entertainment	85.00	83.00
9	sports	namco bandai	85.00	66.50
10	role-playing	falcom corporation	85.00	80.00
11	racing	sony computer entertainment	84.50	87.50
12	first-person	thq	84.00	64.00
13	action	witch beam games	84.00	86.00
14	racing	kunos simulazioni	84.00	85.00
15	action	tecmo	84.00	66.00
16	adventure	aksys games	83.75	84.00

✔ Successfully run. Total query runtime: 54 msec. 280 rows affected.

Insight

(5)

MONTHS WITH THE HIGHEST **EARNING** IN TOURNAMENTS

```
select to_char("month", 'MM') as "month",
concat(round(sum(tournament.earnings)::numeric,2),' mln') as earnings,
sum(tournament.num_events) as "#tournaments"
from tournament_mart.tournament
join tournament_mart.date ON date.id = tournament.date_key
group by to_char("month", 'MM')
order by sum(tournament.earnings) DESC
```

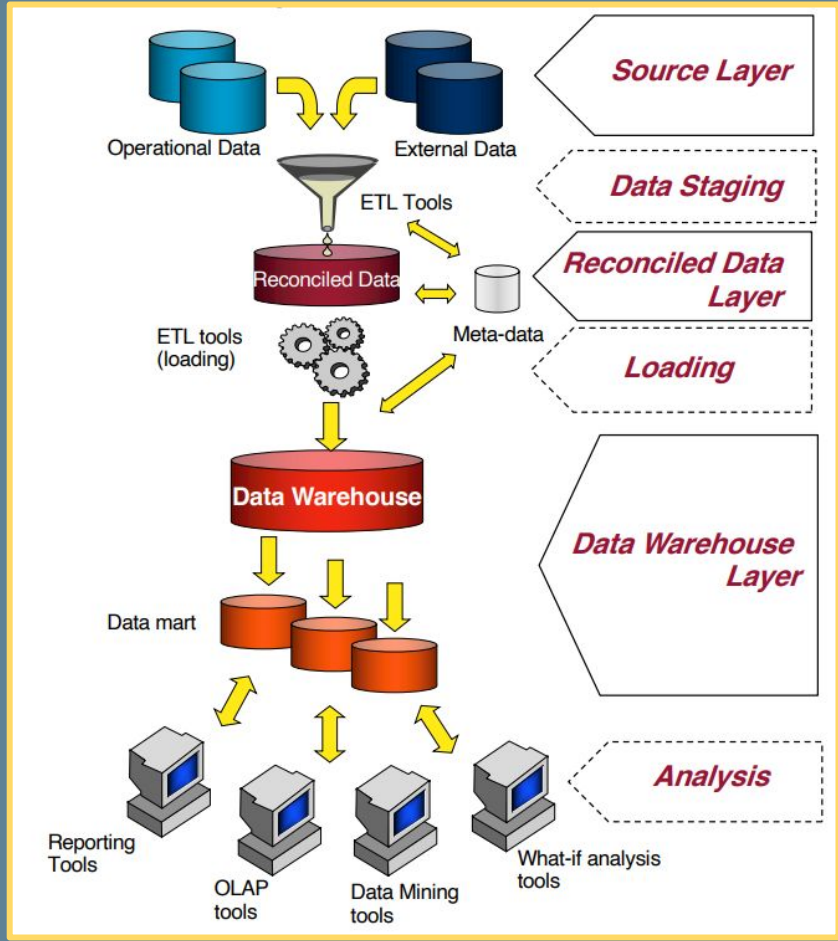
	month text	earnings text	#tournaments numeric
	08	2628042.00 mln	80
	12	2059635.00 mln	110
	11	2010716.00 mln	136
	07	1562222.00 mln	70
	10	1518494.00 mln	102
6	09	1265103.00 mln	38
7	02	836642.00 mln	50
8	01	750453.00 mln	42
9	06	739017.00 mln	81
10	05	653360.00 mln	42
11	03	632653.00 mln	69
12	04	588060.00 mln	96

✔ Successfully run. Total query runtime: 66 msec. 12 rows affected.

Three-Layered DW Architecture

1. Data Sources Collection
2. Data Integration and Reconciliation
   a. Reconciled Schema
   b. ETL processing
3. Data Warehouse
   a. Conceptual Design
   b. Logical Design
   c. Physical Design
4. Analysis

   OLAP, Materialized Views
5. Qlik Sense data mart re-implementation

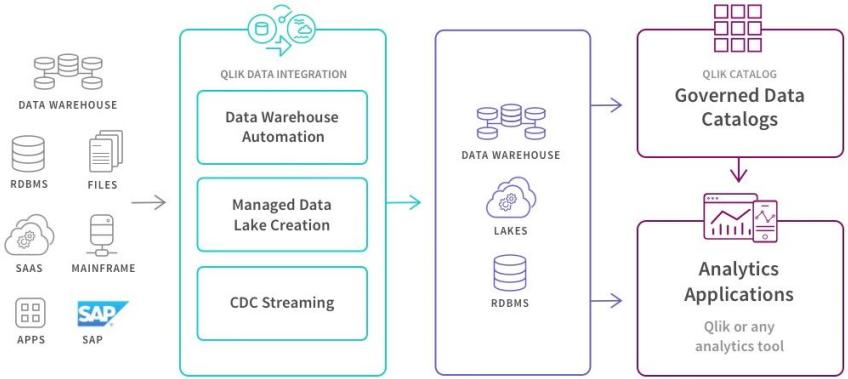


Qlik Sense

Qlik Sense is a smart data analytics platform used in the Business Intelligence field. It's the **evolution of QlikView.**
It is built on top of an *associative analytics engine*, a very powerful alternative of the classical query/based tools.
The platform offers also:

- A sophisticated AI, which permits to give suggests about the associations between tables, the most suitable charts for certain data and even natural-language queries!
- High performance cloud platform, which we used in this project

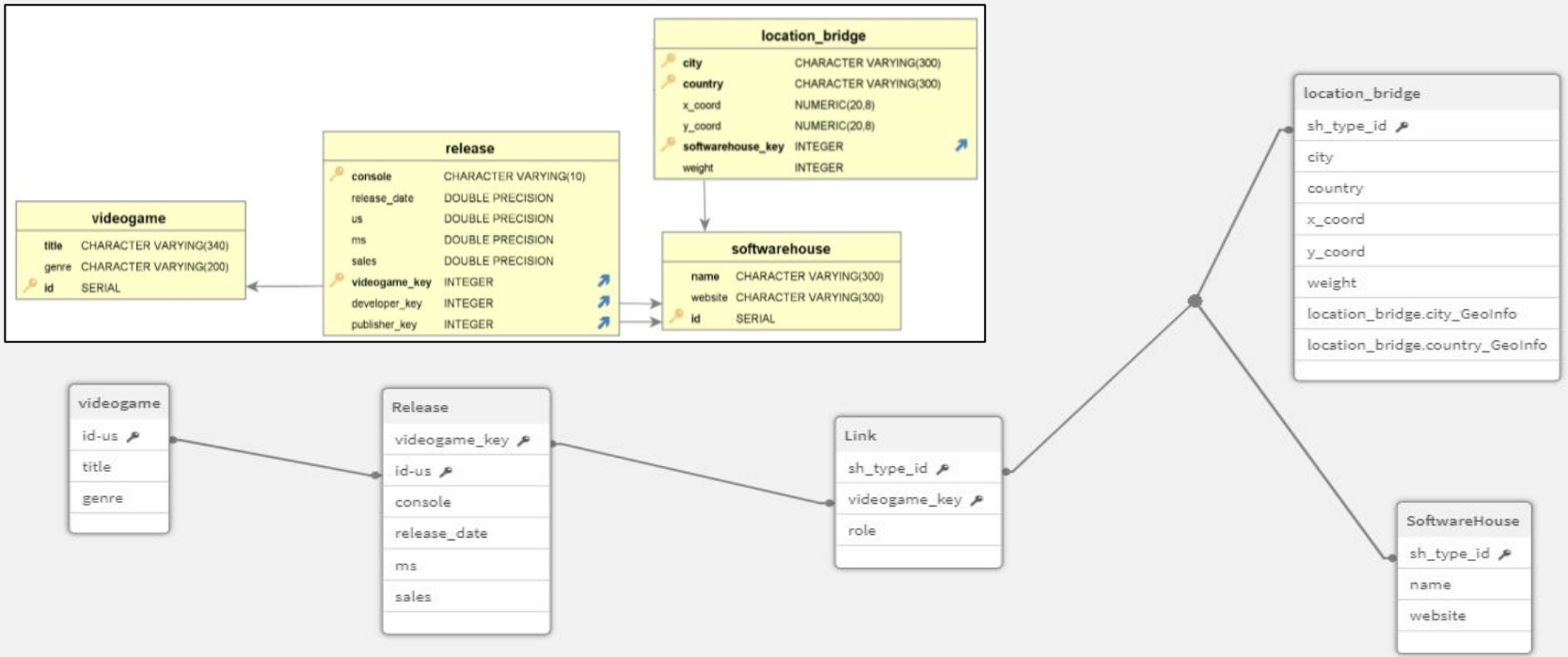
The Qlik suite also offers an entire workflow for manage a data warehouse project



We focused on the re-implementation of our data marts and on the evaluation of the insight presented before

Qlik Sense - Release Data Mart cube

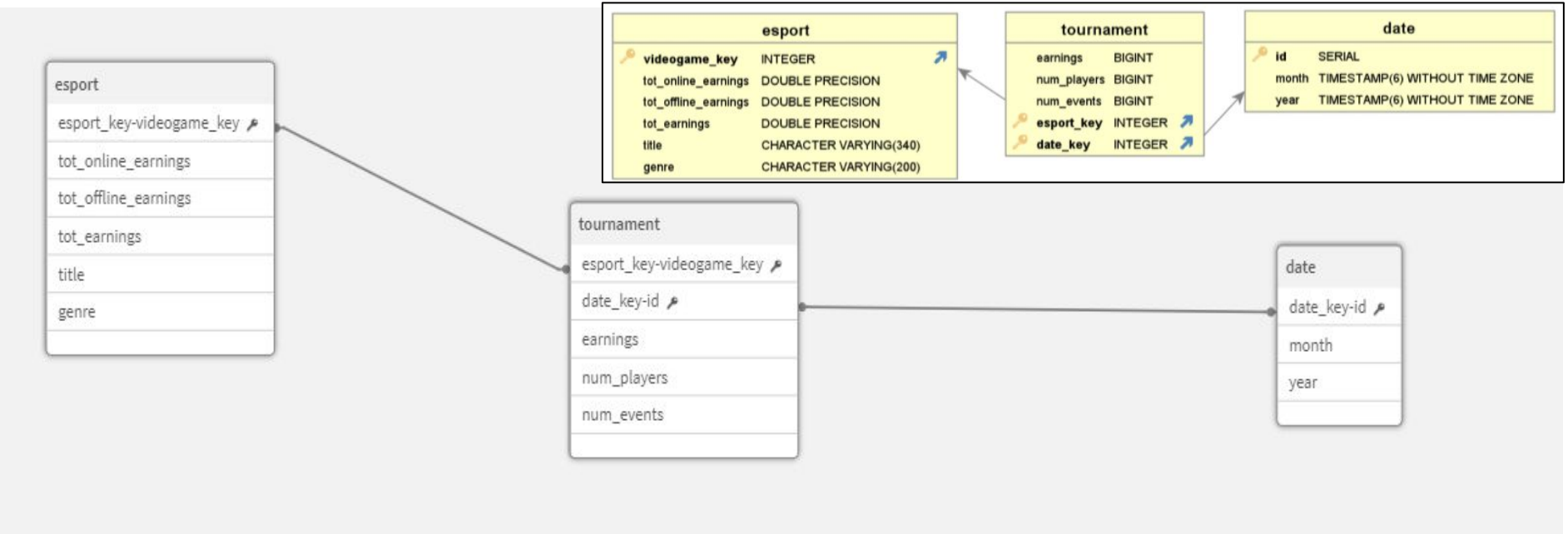
VISUALIZATION of Physical data Model



Qlik Sense does not allow two association with the same table, so we need to convert our data mart creating a link table with the role attribute which contains the values ['Developer', 'Publisher']

Qlik Sense - Tournament Data Mart cube

VISUALIZATION of Physical data Model



In this case we built the same data mart from PostgreSQL, note that the Qlik Sense's AI recognized autonomously both the associations!

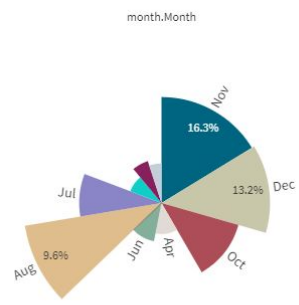
Insight

MONTHS WITH THE HIGHEST EARNING IN TOURNAMENTS

MONTHS WITH THE HIGHEST EARNING IN TOURNAMENTS

month.Year	month.Month	Sum(earnings)	Sum(num_events)
2021	Jan	357000	4
2020	Jul	446154	18
2020	Mar	340368	26
2020	Nov	243522	11
2020	Apr	161597	56
2020	Oct	119858	6
2020	Dec	84627	5
2020	Aug	51432	4
2020	Jan	34080	8
2020	May	26085	5
2020	Jun	12750	26
2020	Feb	2130	3

month.Year	month.Quarter	Sum(earnings)	Sum(num_events)
2021	Q1	357000	4
2020	Q1	376578	37
2020	Q2	200432	87
2020	Q3	497586	22
2020	Q4	448007	22
2019	Q1	275960	14
2019	Q2	502562	25
2019	Q3	392938	20
2019	Q4	1108200	11
2018	Q1	29829	10
2018	Q2	518869	17
2018	Q3	438770	19



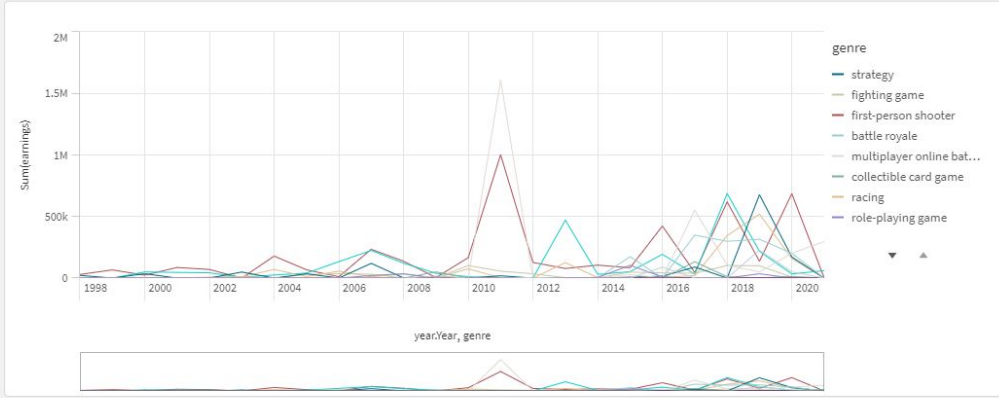
OLAP: Tournament

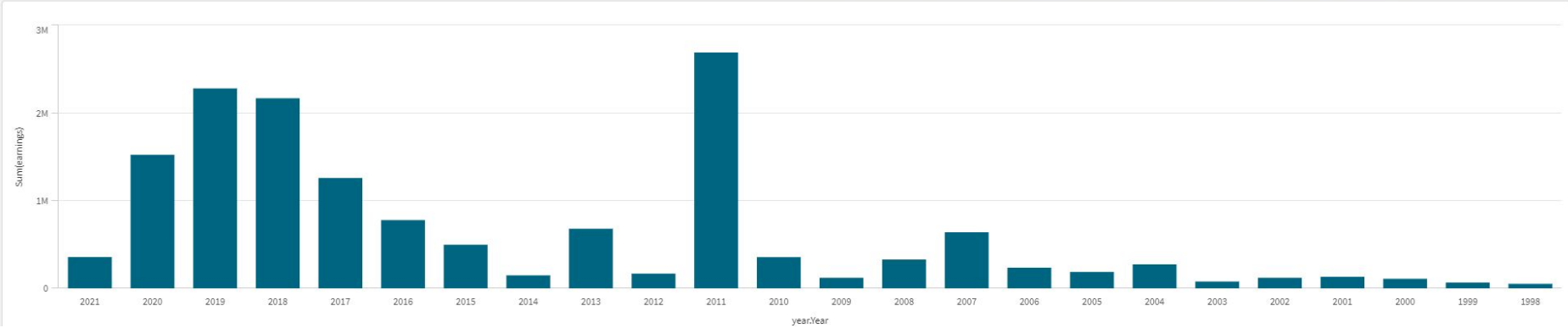
Qlik (3)

Roll up the **TOURNAMENT** multidimensional cube eSport genre and year and **dice** by year:

Roll up the TOURNAMENT multidimensional cube eSport genre and year and dice by year:

genre	Sum(earnings)	num_players	year.Year	Sum(num_events)
multiplayer online battle arena	297000	10	2021	1
sports	60000	34	2021	3
first-person shooter	685415	707	2020	87
battle royale	201000	146	2020	5
multiplayer online battle arena	200600	123	2020	2
strategy	171376	118	2020	15
racing	158209	85	2020	17
third-person shooter	50000	48	2020	2
sports	33827	6	2020	2
collectible card game	13254	7	2020	3
fighting game	7928	101	2020	29
puzzle game	994	23	2020	6
strategy	677436	167	2019	20
racing	519839	28	2019	3
battle royale	315229	178	2019	10





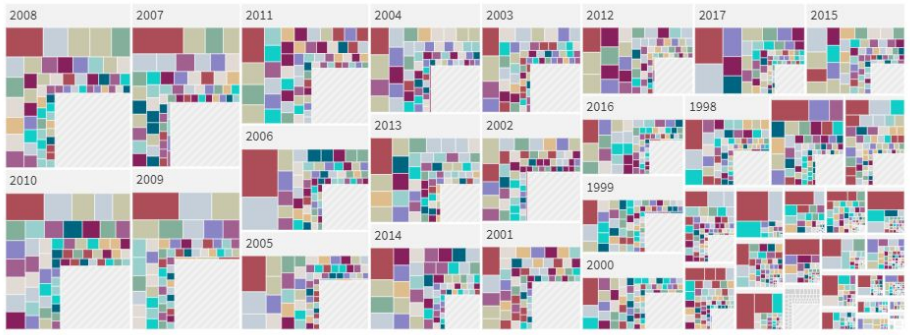
Insight

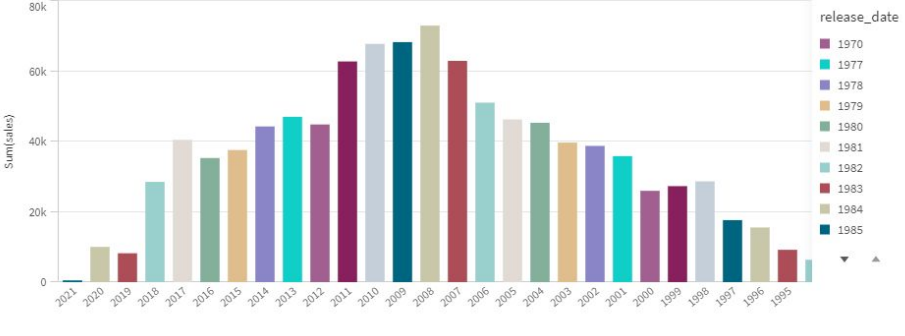
PUBLISHERS WHO EARNED THE MOST BY SALES, PER YEAR

PUBLISHERS WHO EARNED THE MOST BY SALES, PER YEAR

name	if(Sum(sales)>0, Sum(sales))	year
capcom	500	2021
nintendo	5503	2020
sony interactive entertainment	1190	2020
devolver digital	1100	2020
mediatonic	1100	2020
mediatonic ltd.	1100	2020
sucker punch	650	2020
square enix	581	2020
naughty dog	400	2020
blizzard entertainment	370	2020
koei tecmo games	350	2020
intelligent systems	305	2020
konami	250	2020
atlus	230	2020
ndemic creations	200	2020

if(not release_dat... / name	Sum(sales)
2021	500
2020	10027
2019	8214
2018	28527
2017	40474
2016	35306
2015	37578
2014	44284
2013	47029
2012	44847
2011	62770
2010	67761





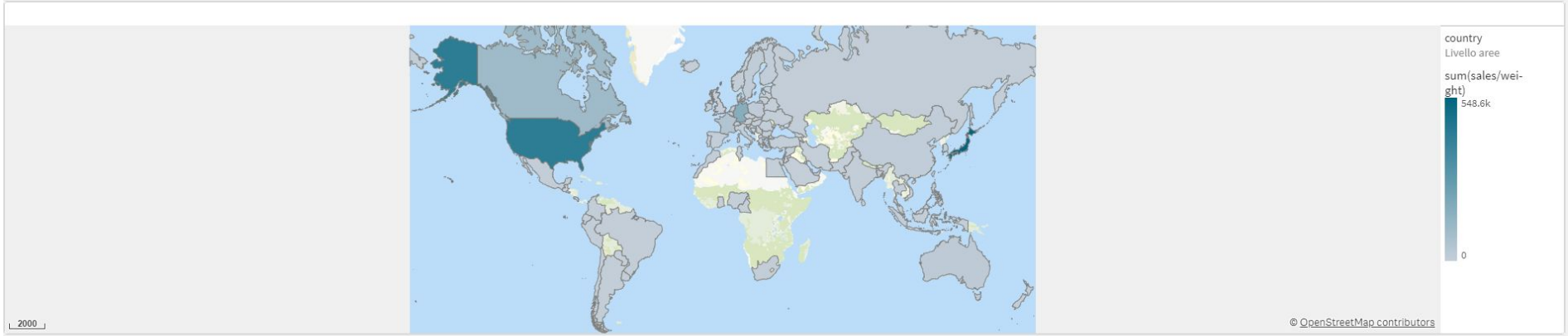
Insight

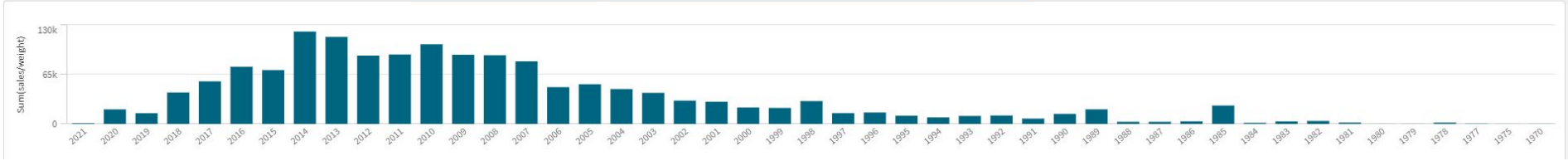
COUNTRIES WHO EARNED THE MOST FROM **SALES PER YEAR**

COUNTRIES WHO EARNED THE MOST FROM SALES PER YEAR

release_date	country	sales	role
2021	japan	250	Publisher
2021	united states	250	Publisher
2020	japan	5534	Publisher
2020	united states	2334.285714...	Publisher
2020	england	200	Publisher
2020	sweden	200	Publisher

release_date	country	sales	role
2021	japan	250	Developer
2021	united states	250	Developer
2020	japan	5083.5	Developer
2020	england	4700	Developer
2020	france	226.4285714...	Developer
2020	sweden	100	Developer



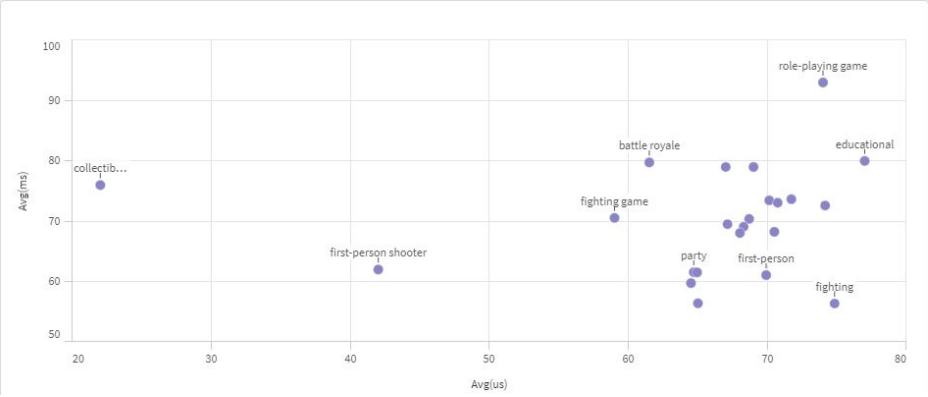


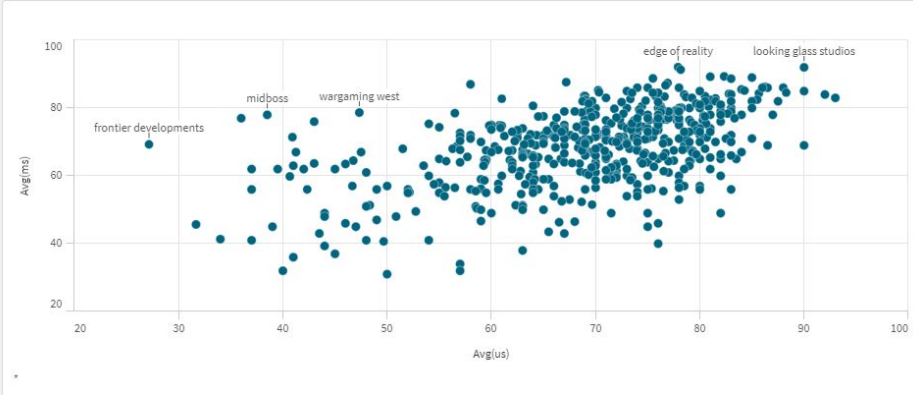
Insight

PAIR **GENRE-PUBLISHER** WITH THE HIGHEST RATINGS

PAIR GENRE-PUBLISHER WITH THE HIGHEST RATINGS

name	genre	avg us	avg ms	role
blizzard entertainment	real-time	92	90	Publisher
square	role-playing	92	84	Publisher
microsoft	real-time	91	92	Publisher
gt interactive	real-time	88	86	Publisher
working designs	role-playing	88	86	Publisher
crave entertainment	sports	88	66.5	Publisher
agetec	simulation	85.5	69.333333	Publisher
bam! entertainment	racing	85	83	Publisher
falcom corporation	role-playing	85	80	Publisher
namco bandai	sports	85	66.5	Publisher
sony computer entertainment	racing	84.5	87.5	Publisher
tecmo	action	84	66	Publisher
sony interactive entertainment	role-playing	84	78	Publisher
microsoft game studios	strategy	83.5	85	Publisher
nintendo	role-playing	83.375	83.125	Publisher





Qlik Sense vs PostGreSQL

QLik Sense:

- Easier to use (high level platform)
- Stunning Ai's support
- Data analytics through charts and dashboards
- Difficult keep track of the engine workflow
- Hidden info about performances

PostgreSQL:

- Complete freedom at query design
- Query optimization with Indexes, and Secondary Views
- Easy to keep track performances
- Harder to use
- No Ai support

CONCLUSIONS

We presented the analysis, design and implementation of a data warehouse, integrating multiple **videogame** industry sources.

At first we gathered all the sources we needed.
Then we extracted, transformed and loaded them with **Pentaho**, reconciling their heterogeneous information.

Finally, we:

- Built a **data warehouse** and two **data marts** on *PostgreSQL*, for reporting and data analysis purposes.
- Replicated the **data marts** on *Qlik Sense*, to produce interactive dashboards and comparing them with previous results.

SAPIENZA
UNIVERSITÀ DI ROMA

Thanks for the attention!

Longo Valerio 1655653
Papi Alessio 1761063

https://github.com/1655653/LSDM