Cloud Data Warehouses

A Data warehouse(DWH) is a system including processes, technologies and data representations that enables us to support analytical processes(business perspective)
A Data warehouse(DWH) is a copy of transaction data specifically structured for query and analysis.
A Data warehouse(DWH) is a subject-oriented, integrated, non-volatile, and time-variant collection of data in support of management's decisions.
3NF => optimised for transactions
Star schema => Optimised for analytics

Data warehouses Architectures

- **Kimball's Bus architecture** => it results in a common dimension model shared by different departments, Data is not kept at the aggregated level but rather at the sub-atomic level, organized by business processes and used by various departments. The design goals are ease of use and query of performance. The ETL process transforms from source to target, conform dimensions, no user query. Applications are ad hoc queries, standard reports, analytic apps.
- Independent data marts => independent data marts are designed by specific business departments to meet their analytical needs.
- **Inmon's corporate information factory(CIF)** this database builds on a 3NF normalized database and then allows documented data denormalization for data marts. There are two ETL processes, source systems -> 3NF databases, 3NF databases to departmental data mart. It is dimensionally modeled and aggregated.
- Hybrid Bus and CIF => there are people who combine the Kimball model and the Inmon model, The Hybrid Kimball Bus and Inmon CIF model stays true to the Enterprise Data Warehouse with data maintained in 3NF even though normalized data tables may not be optimal for BI reports.

An OLAP cube is an aggregation of a fact metric on different dimensions, common OLAP operations include roll-up, drill-down, slice and dice. OLAP cubes should store the finest grain of data, sometimes called atomic data.
Roll-Up => Sum up numbers to get less values
Drill-down => decompose the sales in each city to smaller districts.
Slice => is to drop dimensions and N dimensioned cube to an N - 1 dimensioned cube.
Dice => creating a sub-cube by restricting the number of values.
A 3D cube usually has 4 columns, so we can generally agree that an ND cube usually has N -1 columns

OLAP cubes technology

- Pre-aggregate the OLAP cube and save them on a special purpose, non-relational database (MOLAP)
- Compute the OLAP cube on the fly from existing relational databases where the dimensional model resides (ROLAP)

AWS for Data warehousing

When using AWS you can either go Cloud-managed (Amazon RDS(SQL or NoSQL), Amazon DynamoDB(SQL Columnar massively parallel), Amazon S3), self-managed (EC2 + PostgreSQL, EC2 + Cassandra, EC2 + Unix FS)
Amazon Redshift

- Column-oriented storage
- Best suited for storing OLAP workloads
- Internally it is modified Postgres
- It is a cloud-managed MPP(Massively Parallel Processing => the idea of this is to parallelize the execution of a single query on multiple machines) database

Redshift actually implements a Cluster, that Cluster contains one leader node and one or more compute nodes. The leader node co-ordinates the compute nodes, handles external computation, and optimizes query execution. The compute nodes, each has its own CPU, storage and disk. Each compute node is logically divided into a number of slices, a cluster with n slices can compute n-partitions of a table simultaneously. The number of nodes in a cluster equals the number of EC2 instances in that cluster. To move data from S3 buckets to Redshift use the COPY command, if the file is really large it is best to break it into small parts to enable you to COPY them in parallel. For very small data, you can ingest it directly to the S3 machine.

NOTE: You need to create am IAM role to be able to ingest data to SQL from Redshift, also to access your DWH from the outside you need to open a TCP port.
TODO => Exercise 1 Number 23
There are two types of nodes in AWS, the storage-optimized nodes(ds), compute-optimized nodes(dc).
You can configure an ETL server to communicate between databases.
Scale-up => get the most powerful nodes.
scale-out => get more medium-sized nodes.
In optimizing table design the two possible strategies are Distribution style, Sorting Key.

DIstribution style

- EVEN distribution
- AUTO distribution
- ALL distribution
- KEY distribution

Two ways to eliminate shuffling, we either do the EVEN and ALL on the fact and dimension tables respectively or if the dimension tables have lots of rows we can do the KEY and KEY. You can define s sorting key to minimize the Query time. Usually, put a dist key on the big tables, but then put a sort key on all the other tables.