Week 3 Data Warehouse

- Data Warehouse.
- partitonning and clustering.
- Best practices.
- Internals Big Query.
- Big Query Machine learning.
- Deployment of BQ ML Models.
- Doing tasks using Airflow

	OLTP	OLAP
purpose	- Online Transaction Processing - Control and run essential business operations in real time	Online Analytical Processing Plan, solve problems support Decissions, Discover insights
Data update	Short, fast updates initialized by user.	Data periodicaly refreshed with scheduled, long run batch job
DB design	Normalized DB For Efficiency For fast responce with user	Denormalized DB for analysis
Size	Generally small if historical data archived	Generally large due to aggregating large DB Also to run Analytic need a data
Backup productivity	Regular Backup happen increase productivity user	Lost data can loaded from OLTP increase productivity for bussiness mnger

Data warehouse (DW)

- DW is an OLAP solution meant for reporting and data analysis, unlike OL which flow ELT DW flow ETL

- DW receives data from different data sources, which then processed in staging area before being ingested to actual warehouse (DB), DW may then feed data to separat Data Marts "smaller DB which end users can use"

Big Query (BQ)

- BQ is a DW solution offered by GCP.
- BQ is different than GCS
  - BQ is serrerless: no server to mange or DB software to install
  - BQ is scalable and high availabilty: Google take care of underling Software and infrastmethe.
  - BQ has Built in feathes: - Machine learning
    - Geospatial analysis
    - Business intelligence
  - BQ maximize flexability by separating data to diffirent engines.

Alternative AWS Redshift, Azure Synapse

Pricing

GCP charges you in most of feathes like:
- processing
- ingestion
- storage .02 $/GB/month
- extraction

Data processing:

- on demond [Default]: 5$/TB/month [first TB free]
- Flat rate:
  2000 $/month
  cost is not with data, But with slots (cpu)
  minimum 100 slots for 2k $/M.
  But if you run a lot of queries and 100 slot not enough queries must wait till free slot.
  on demond dosen't have this issue as you deal with storage not slots.
- Flate rate limited with queries due slots.
  on demond limited with storage yo process.

- when we run query on BQ on the ▫ top right estimated values for [cost & time appear]

→ External tables:
BQ supports external data sources, you may query these sources directly from BQ, even data isn't in BQ.

partations

BQ tables can be partitioned into multiple smaller tables, Filter based on data, so we can query a specific sub-table based on data we are interested in.

- partation improve: -performance -reduce cost
- tables are partitioned based on
  - Time stamp. -Data.
  - Any numerical values.
  - partation may be [daily, month, year] as a value for partation default daily.
- BQ limit number of partations to 4000/table
  - Example run Query on partationed and non partationed
    - partationed 106 MB of Data to process
    - Non partationed 1600 MB // //
    - ~~only one~~ column cond used

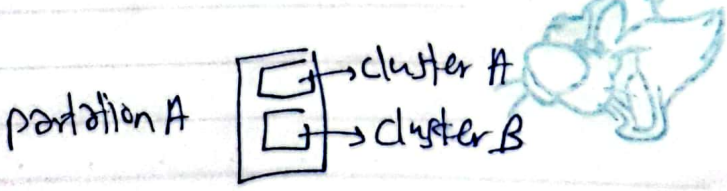
Clustering

Rearranging table based on it's values ex categorical we can cluster multible column till (4).

- values of categorical can be any thing string or even numerical.
- Tables with less than 1 GB (dosen't show significant improvement, It may cost and increase time ....!

clustering	partitioning
cost benefit unknown BQ Can't estimate cost "for example we may have Query with clusters from 4 columns each one could have X clusters so time consuming to estimate.	cost known upfront. Estimate cost before run "for ex we know that we have only one column and limit of 4000 partations so limited to estimate.
High granularity multiple criteria can used to sort	low granularity only single column, often time.
clustered fixed place	partations are flexable can be added.
Good for queries that uses filters or aggregation (تجميع)	Good when filter based on single column.

- use of clustering if:
  - partations > 4000.
  - partation results small amount of data/partation.
- BQ has automatic re [clustering, partationg] when we add, delet data.
- For partationed data we may have clustering inside partation.



Best Practices

1. Cost reduction
   - Avoid SELECT * as reduce amount of processed data would reduce cost.
   - price your queries
   - use clustered and partationed.
   - use streaming inserts with caution.
   - Materialize query results. "Memorize it" Cache

2. Query performance
   - Filter on partationed columns.
   - Denormalize data....?
     - ~~Normalization~~ is very important to remove any ~~redundency~~ in the data.
     - Make relation between data.
     - store data in an atomic form [First, Mid, last]
     - So why we do Denormalization ...?
       - Normalized data uses Joins to keep relationship between columns, But Joins takes alot of resources and reduce performance of query.

Ex Normalized : Bringing Breakfast, launch, dinner from suber market, this Subermarket store each element alone, so you start with break fast an go through all element section and so on.

Denormalized: Bringing some thing, but here subermarket sections for meal section for launch and so on, So you won't go to all sections.

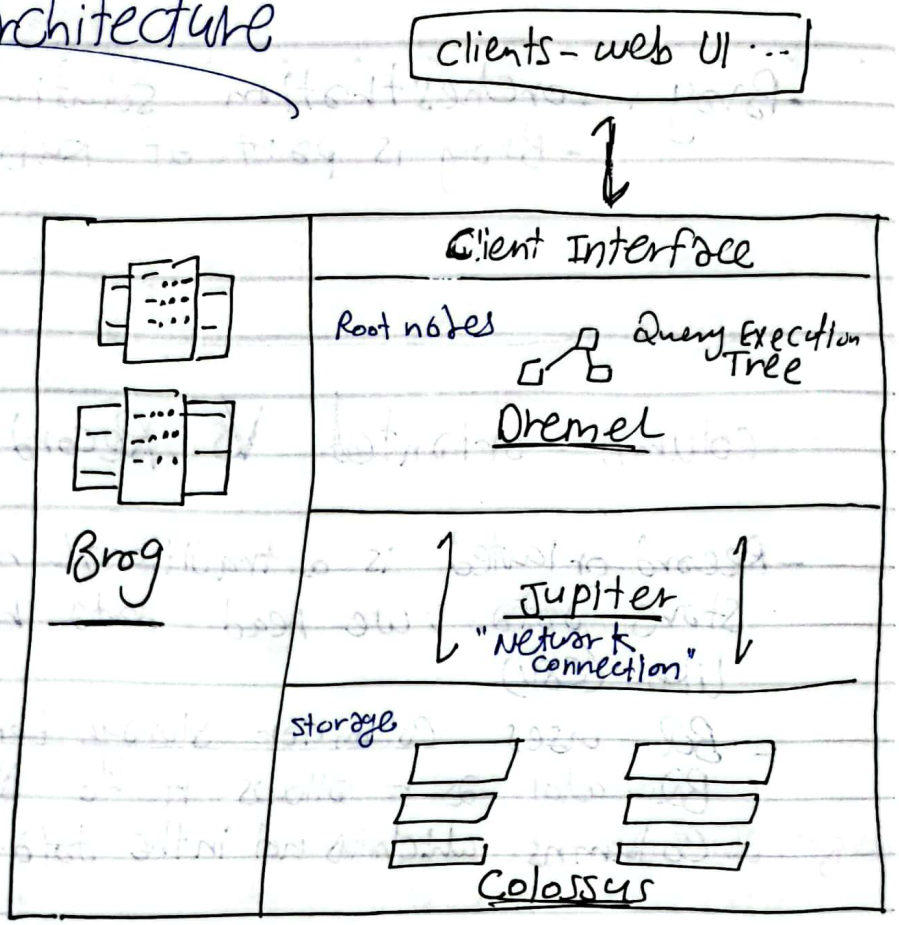
* missing some concepts in tricks

For this example
- user would only take few minutes as every thing is organised by meal.
- we may have some objects repeated on different places like bread on [dinner, breakfast]

For supermarket
- that would reduce their speed and performance as we have redundency bread repeated on different places.

- use nested or repeated columns
- External resources reading from bucket affect performance badly.
- Reduce data before joins.
- Don't threat [with] clauses as prepared statment.
- Avoid oversharding tables ["storing data in Multible tables"]
- Avoid java-Script user-defined functions.
- use appoximate aggregation function.
- order statement should be last part.
- start with largest tabel and small....

Big Query Architecture



Colossus: -Google global storage with speed ITB/S using Jupiter.
- Columnar storage Format & Compression algorithms to store data.
- Handel replication, replace and recovery and disthupted managent.

Jupiter: -Network to connect dremel and Colossus in house network to interconnect database.

Dremel: [Brain]
- Compute part of BQ
- SQL Queries to tree
- leaves of tree slots branch mixers
- slots reading data from storage
- mixers perform aggregation.
- Dremel split slots to queries as needed.

- Borg : - orchestration solution
  - Borg is part of kubernetes.

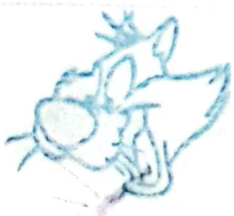
---

column - oriented Vs Record - oriented

- Record oriented is a traditional method to store data, we read data row by row like (CSV)
- BQ uses Columnar storage Format, this help BQ alot as it allows it to discard(تجاهل) other columns which is not in the data.

- Dremel modify queries to an execution tree parts of queries assigned to mixers then to even small parts to slots to access colossus.

- Colum storage allows Fast retrieval (استرجاع) From colossus which return to mixer to perform aggregation.



ML with BQ

BQ-ML allows:
- Run and execute Machine learning models.
- All of that using SQL only without of any experience with Python.
- No need for experts just run SQL then:
  - feature Eangineering Auto..
  - Train of the model.
  - Export & Deploy of model.

- features like:
  - predict values - linear regression
    - AutoML, DNN, Wide regressor
  - Identify spam emails - logestic regression
    - boosted Trees, DNN
  - Generate reccomendation.
  - Reduce data dimenstionality.
  - find anomalies [Detect fraud, credit risk]
  - Group data to clusters [customer segmentation]
  - Time series forcasting [ARIMA]

- we will creat New table which contain certin columns casted.

CREATE OR REPLACE Model 'taxi-rides-ny.tip-model'
options(
  model_type = 'linear_regression'
  input_label_cols = ['tip_amount'], → Target Label
  DATA_SPLIT_METHOD = 'AUTO_SPLIT') AS
SELECT * FROM ___ data set

32 show all details of model like:
- Graphs of train.
- Evaluation metrics.
- Model Graph.

SELECT * FROM ML.FEATURE_INFO
(MODEL 'taxi_tip_model');

like describle

- ML.FEATURE
- ML.EVALUATE
- ML.PREDICT
- ML.Explain_predict
  Return prediction with column
  predicted = 1.321 ← fare drop of pickup
  - struct(3 as top_k_feature)

- Hyper parameter is available

BQ ML Deployment :

- we will Export model use Tensorflow serving which is docker img and use HTTP to get prediction

- steps :
  1. Authenticate GCP
  2. Export model to Cloud Storage Bucket.
  3. Download Exported model.
  4. Run docker image.
  5. with image working, run prediction

Curl \ ← Features
-d '{"instances": [{" passager_count":1,"trip_distance":12.2...}]}' \
-X POST http://localhost:8501/v1/models/tip_model:predict

returned value is preidection.

Integration BQ with Airflow

- we will use airflow to automate process of download files and pocess it each certin time using Cron Job
  1. GCS - 2 - GCS [moving Files between folders]
  2. GCS - 2 - bq-ext [GCS to BQ Extend]
  3. bq - 2 - partotined

- we will use Google-provided operators

. we extend data in GCS for easier processing.