RDBMS Modelling - SQL

Creating the Date table

```
CREATE TABLE dbms-sjsu.date.date AS
SELECT
  FORMAT_DATE('%F', d) as id,
  d AS full_date,
  EXTRACT(YEAR FROM d) AS year,
  EXTRACT(WEEK FROM d) AS year_week,
  EXTRACT(DAY FROM d) AS year_day,
  EXTRACT(YEAR FROM d) AS fiscal_year,
  FORMAT_DATE('%Q', d) as fiscal_qtr,
  EXTRACT(MONTH FROM d) AS month,
  FORMAT_DATE('%B', d) as month_name,
  FORMAT_DATE('%w', d) AS week_day,
  FORMAT_DATE('%A', d) AS day_name,
  (CASE WHEN FORMAT_DATE('%A', d) IN ('Sunday', 'Saturday') THEN 0 ELSE 1 END) AS
day_is_weekday,
  FORMAT_DATE('%Y%m%d', d) as date_yymmdd
FROM (
  SELECT
        *
  FROM
        UNNEST(GENERATE_DATE_ARRAY('2023-01-01', '2030-01-01', INTERVAL 1 DAY)) AS d )
```

Transform Codes

1. **Service**

   ```
   create table dbms-sjsu.BART_transform.service as
   select
   s.service_id,monday,tuesday,wednesday,thursday,friday,saturday,sunday,d1.full_date as
   start_date,d2.full_date as end_date
   from `dbms-sjsu.BART.service`s
   left join `dbms-sjsu.date.date` d1
   on CAST(s.start_date as string)=d1.date_yymmdd
   left join `dbms-sjsu.date.date` d2
   on cast(s.end_date as string)=d2.date_yymmdd
   ```

2. **Calendar_dates**

create table dbms-sjsu.BART_transform.calendar_dates as
select cd.service_id,d1.full_date as date, d1.date_yymmdd
from dbms-sjsu.BART.calendar_dates cd
left join `dbms-sjsu.date.date` d1
on cast(cd.date as string) = d1.date_yymmdd

3. **Riders Fare**

```
create table dbms-sjsu.BART_transform.riders_fare as
select
fr.price,rc.rider_category_description,f.origin_station,f.destination_station
from `dbms-sjsu.BART.fare_rider_categories` fr
left join `dbms-sjsu.BART.rider_categories` rc
on fr.rider_category_id=rc.rider_category_id
left join `dbms-sjsu.BART.fare_rules` f
on fr.fare_id=f.fare_id
order by origin_station
```

4. **Current Day active schedules**

create table if not exists dbms-sjsu.BART_transform.service_current_day as
select service_id, CASE WHEN current_date BETWEEN start_date and end_date then 1
else 0 end current_day_schedule
from dbms-sjsu.BART_transform.service
group by service_id,start_date,end_date

5. **Next Stop details for the routes in different trips**

create table dbms-sjsu.BART_transform.next_stop as
select *,
LEAD(stop_name) OVER(PARTITION BY service_id,trip_id,route_id,direction_id
ORDER BY stop_sequence) as next_stop
From
(select t.trip_id,t.service_id,r.route_id, r.route_short_name,
d.direction_id,d.direction,st.stop_id,s.stop_name,stop_sequence
from dbms-sjsu.BART.stop_times st
left join dbms-sjsu.BART.trips t
on st.trip_id=t.trip_id
left join dbms-sjsu.BART.directions d
on t.route_id=d.route_id and t.direction_id=d.direction_id

```
left join dbms-sjsu.BART.routes r
on d.route_id=r.route_id
left join dbms-sjsu.BART.stops s
on st.stop_id=s.stop_id
order by trip_id,stop_sequence) a
```

6. **Most connected Station through the highest number of trips.**

```
create view dbms-sjsu.BART_model.most_connected_station as

with active_service as
(select * from dbms-sjsu.BART_transform.service_current_day
where current_day_schedule=1),

main as (select ns.*
from dbms-sjsu.BART_transform.next_stop ns
inner join active_service ass
on ns.service_id=ass.service_id)

select stop_name,count(distinct trip_id) as distinct_trips
from main
group by stop_name
order by count(distinct trip_id) DESC
```

7. **Stations with highest number of functioning routes**

```
create view dbms-sjsu.BART_model.functional_routes as

with active_service as
(select * from dbms-sjsu.BART_transform.service_current_day
where current_day_schedule=1),

main as (select ns.*
from dbms-sjsu.BART_transform.next_stop ns
inner join active_service ass
on ns.service_id=ass.service_id)

select stop_name,count(distinct route_id) as distinct_routes
from main
group by stop_name
order by count(distinct route_id) DESC;
```

8. **Routes which connect the highest number of stations**

```
create view dbms-sjsu.BART_model.routes_stops as
```

```
with active_service as
(select * from dbms-sjsu.BART_transform.service_current_day
where current_day_schedule=1),
main as (select ns.*
from dbms-sjsu.BART_transform.next_stop ns
inner join active_service ass
on ns.service_id=ass.service_id)

select route_short_name,count(distinct stop_id) as distinct_stops
from main
group by route_short_name
order by count(distinct stop_id) DESC;
```

9. **Next Arrival Time for bart across different trips**

```
create table `dbms-sjsu.BART_transform.next_arrival_time` as
with main as (select
stop_id,st.trip_id,st.arrival_time,st.departure_time,r.route_short_name,d.direction,
st.stop_sequence, LEAD(arrival_time) OVER(PARTITION BY
st.trip_id,d.route_id,d.direction_id ORDER BY stop_sequence) as next_arrival_time
from dbms-sjsu.BART.stop_times st
left join dbms-sjsu.BART.trips t
on st.trip_id=t.trip_id
left join dbms-sjsu.BART.directions d on t.route_id=d.route_id and
d.direction_id=t.direction_id
left join dbms-sjsu.BART.routes r on d.route_id=r.route_id
order by trip_id) select * from main
```

10. **Avg time between trains across stations for a particular route**

```
create view dbms-sjsu.BART_model.route_avg_wait_times as
select route_short_name,direction,stop_id,stop_sequence,sum(time_diff_mins) as
time_diff,count(distinct trip_id) as trips,
COALESCE(sum(time_diff_mins)/count(distinct trip_id),0) as avg_wait_time
From
(select route_short_name,direction,stop_id,trip_id, stop_sequence,
TIME_DIFF(next_arrival_time,departure_time, MINUTE) as time_diff_mins
from `dbms-sjsu.BART_transform.next_arrival_time`)
group by route_short_name,direction,stop_id,stop_sequence
```

11. **To display the category rider level price between stations**

```
select * from dbms-sjsu.BART_transform.riders_fare
```

12. **All the direct VTA routes to and from Berryessa VTA Stop**

```
create view dbms-sjsu.VTA_model.to_fro_berryessa as
with main as (select
st.arrival_time,st.departure_time,ra.route_short_name,direction_id,s.stop_name
from dbms-sjsu.VTA.stop_times st
left join dbms-sjsu.VTA.trips t
on st.trip_id=t.trip_id
left join dbms-sjsu.VTA.stops s
on st.stop_id=s.stop_id
left join dbms-sjsu.VTA.route_attributes ra
on t.route_id=ra.route_id
where lower(stop_name) like ('%berryessa%'))

select route_short_name,stop_name,sjsu_tag,arrival_time,departure_time
from (select *,
CASE WHEN direction_id = "1" THEN 'towards SJSU'  else 'from SJSU' end as sjsu_tag
from main)
```

13. **If we consider Santa Clara 6th to be the San Jose Bus Stop, then the below query gives us all the direct routes that are available from San Jose.**

```
Create view dbms-sjsu.VTA_model.from_sjsu as
with main as (select
st.arrival_time,st.departure_time,ra.route_short_name,t.direction_id,s.stop_name,d.directi
on
from dbms-sjsu.VTA.stop_times st
left join dbms-sjsu.VTA.trips t
on st.trip_id=t.trip_id
left join dbms-sjsu.VTA.stops s
on st.stop_id=s.stop_id
left join dbms-sjsu.VTA.route_attributes ra
on t.route_id=ra.route_id
left join dbms-sjsu.VTA.directions d
on ra.route_id=d.route_id and CAST(t.direction_id as int)=d.direction_id
where lower(stop_name) like ('%santa clara & 6th%')
or lower(stop_name) like '6th & santa clara')

select * from
        (select concat(route_short_name," ",direction) as
route_name,stop_name,arrival_time,departure_time
```

from main) a

14. Fare prices from different BART station to Berryessa

```
create view dbms-sjsu.BART_model.fare_to_sjsu as
with main as (select origin_station,destination_station,
SUM(CASE WHEN rider_category_description = 'Senior/Disabled Clipper' THEN price else 0 end) as senior_disabled_clipper,
SUM(CASE WHEN rider_category_description = 'Youth Clipper' THEN price else 0 end) as youth_clipper,
SUM(CASE WHEN rider_category_description = 'Clipper START' THEN price else 0 end) as clipper_start
from dbms-sjsu.BART_transform.riders_fare
where destination_station like '%erryessa%'
group by origin_station,destination_station),

clipper as (
 select sd1.station_name as origin_station,sd2.station_name as destination_station, price
from dbms-sjsu.BART.fare_rules fr
left join dbms-sjsu.BART.station_details sd1
on fr.origin_id=sd1.station_code
left join dbms-sjsu.BART.station_details sd2
on fr.destination_id=sd2.station_code),

main2 as (select main.*,clipper.price as clipper_price
from main left join clipper
on main.origin_station=clipper.origin_station
and main.destination_station=clipper.destination_station)
select * from main2
```

15. Updated BART price details if the Phase 2 is implemented

```
create view dbms-sjsu.BART_model.phase2_prices_to_sjsu as
select origin_station,destination_station,
(youth_clipper+2.50) as ph2_youth_clipper,
(senior_disabled_clipper+2.06) as ph2_senior_disabled_clipper,
(clipper_start+4.42) as ph2_clipper_start,
(clipper_price+5.50) as ph2_clipper_price
from dbms-sjsu.BART_model.fare_to_sjsu
Group by origin_station,destination_station
```

16. Regular BART price details till Phase 2 is not implemented.

```
create view dbms-sjsu.BART_model.wop2_prices_to_sjsu as
select origin_station,destination_station,
(youth_clipper+2.50) as ph2_youth_clipper,
(senior_disabled_clipper+2.5) as ph2_senior_disabled_clipper,
(clipper_start+2.5) as ph2_clipper_start,
(clipper_price+2.5) as ph2_clipper_price
from dbms-sjsu.BART_model.fare_to_sjsu;
```

17. Query which we are using to test the performance of Graph & RDBMS

```
with main as (select origin_station,destination_station,
        SUM(CASE WHEN rider_category_description = 'Senior/Disabled Clipper' THEN
price else 0 end) as senior_disabled_clipper,
        SUM(CASE WHEN rider_category_description = 'Youth Clipper' THEN price else 0
end) as youth_clipper,
        SUM(CASE WHEN rider_category_description = 'Clipper START' THEN price else 0
end) as clipper_start
from dbms-sjsu.BART_transform.riders_fare
where destination_station like '%erryessa%'
group by origin_station,destination_station),

clipper as (
 select sd1.station_name as origin_station,sd2.station_name as destination_station, price
from dbms-sjsu.BART.fare_rules fr
 left join dbms-sjsu.BART.station_details sd1
  on fr.origin_id=sd1.station_code
 left join dbms-sjsu.BART.station_details sd2
on fr.destination_id=sd2.station_code),

main2 as (select main.*,clipper.price as clipper_price
from main left join clipper
on main.origin_station=clipper.origin_station
and main.destination_station=clipper.destination_station)
```

```
select origin_station,destination_station,
(youth_clipper+2.50) as ph2_youth_clipper,
(senior_disabled_clipper+2.06) as ph2_senior_disabled_clipper,
(clipper_start+4.42) as ph2_clipper_start,
(clipper_price+5.50) as ph2_clipper_price,
(youth_clipper+2.50) as woph2_youth_clipper,
(senior_disabled_clipper+2.50) as woph2_senior_disabled_clipper,
(clipper_start+2.50) as woph2_clipper_start,
(clipper_price+2.50) as woph2_clipper_price
from main2
```