Physical DW Design

1. Queries

First KPI:

```
SELECT TD.monthid, Sum(AU.flighthours) AS FH, Sum(AU.flightcycles) AS FC
FROM aircraftutilization AU, temporaldimension TD, aircraftdimension AD
WHERE AU.aircraftid = AD.id AND AU.timeid = TD.id AND AD.model = '777'
GROUP BY TD.monthid
ORDER BY TD.monthid
```

Second KPI:

```
SELECT M.Y, Sum(AU.scheduledoutofservice) AS ADOSS, Sum(AU.unscheduledoutofservice) AS ADOSU
FROM aircraftutilization AU, months M, temporaldimension TD
WHERE M.id = TD.monthid AND AU.timeid = TD.id AND AU.aircraftid = 'XY-WTR'
GROUP BY M.y;
```

Third KPI:

```
SELECT LR.monthid,  1000 * ( marep + pirep ) / fh AS RRh,  100 * ( marep + pirep ) / fc AS RRc,  1000 * pirep /
fh AS PRRh, 100 * pirep / fc AS PRRc, 1000 * marep / fh AS MRRh,
100 * marep / fc AS MRRc
FROM (SELECT TD.monthid,
        Sum(AU.flighthours) AS FH, Sum(AU.flightcycles) AS FC
        FROM temporaldimension TD, aircraftdimension AD, aircraftutilization AU
        WHERE AU.aircraftid = AD.id AND AU.timeid = TD.id AND AD.model = '777'
        GROUP BY TD.monthid) AU,
        (SELECT L.monthid,  Sum(CASE WHEN PD.role = 'M' THEN L.counter ELSE 0 END) AS MAREP,
                            Sum(CASE WHEN PD.role = 'P' THEN L.counter ELSE 0 END) AS PIREP
        FROM logbookreporting L, aircraftdimension AD, peopledimension PD
        WHERE L.aircraftid = AD.id AND PD.id = L.personid AND AD.model = '777'
        GROUP BY L.monthid) LR
WHERE AU.monthid = LR.monthid;
```

Fourth KPI:

```
SELECT LR.model, 1000 * marep / fh AS MRRh, 100 * marep / fc AS MRRc
FROM    (SELECT AD.model, Sum(AU.flighthours) AS FH, Sum(AU.flightcycles) AS FC
        FROM aircraftdimension AD, aircraftutilization AU
        WHERE AU.aircraftid = AD.id
        GROUP BY AD.model) AU,
        (SELECT AD.model, Sum(CASE WHEN PD.role = 'M' THEN L.counter ELSE 0 END) AS MAREP
        FROM logbookreporting L, aircraftdimension AD, peopledimension PD
        WHERE L.aircraftid = AD.id AND PD.airport = 'KRS' AND PD.id = L.personid
        GROUP BY AD.model) LR
WHERE AU.model = LR.model;
```

2. Assumptions and Data structures

We created the following four data structures to optimize the DW for the given workload:

1. **Bitmap join index for AircraftUtilization on AircraftDimension.Model**

This kind of index is only useful for join tables with low cardinality columns, in this case, the indexed column is the attribute model of the table AircraftDimension that has a cardinality of 8. To be able to create this structure, we firstly added a unique constraint to the dimension table AircraftDimension on the attribute ID. Then, the bitmap index creation. This data structure was created to improve the first KPI, as by having it we are in some way materializing joins in advance and the access to the dimension table could be avoided. Thus, this would make the filtering by aircraft model faster.

2. **Bitmap Index on table AircraftUtilization on the attribute AircraftId.**

   This data structure was created to improve the second KPI. The attribute AircraftIdis is suitable to be a bitmap index because it has 20 distinct values, and each value has on average 5000 repetitions. With this we make the selection of an aircraft of the fleet faster.

3. **Bitmap Join Index for LogbookReporting on AircraftDimension.Model.**

   This data structure was created to improve the third KPI. Same as before, the aim is to pre-join the tables by indexing LogbookReporting, but the attribute we use for indexing is actually from the table AircraftDimension. Thus, the access to the dimension table is avoided and this would make the filtering by aircraft model faster.

4. **Bitmap Join Index for LogbookReporting on PeopleDimension.Airport**

   To be able to create this structure, we added a unique constraint to the dimension table PeopleDimension on the attribute ID. This data structure was created to improve the fourth KPI. The attribute airport was chosen to be indexed column because it's the filtering value, and it has a cardinality of 20 and each value has on average 16500 repetitions. By doing so, the selection of rows that fulfill the filtering criteria would be faster, as only 20 lists (or even less if the bitmap is compressed) are needed to be checked.

The star-join transformation that enables query rewrite was tried for the performance improvement, but it did not help as some queries do the joins between dimensions tables and they do not have in the select clause only attributes from fact tables.
B-tree index was not used because in general they are more suitable for high cardinality columns and creating a b-tree index for them is expensive, so due to the disk space constraint it was not possible.