Silvia Miralles
silvia.miralles
Alejandro Dapena
alejandro.dapena

ÍNDEX:

Introduction with a brief summary.

La fórmula 1 celebra el seu 70 aniversari aquest any. Per celebrar-ho, volen renovar la seva base de dades, pensant en nous format (com el format OLAP).

La finalitat d'aquesta pràctica és, mitjançant un codi en Java (que es conecta tant a la BBDD del servidor com a la local), copiar la base de dades de Fórmula 1 emmagatzemada en el servidor Puigpedrós i copiar-la en dos bases de dades locals. Una d'aquestes bases de dades serà una copia exacta de la bases de dades del servidor mentre que l'altre serà una base de dades no normalitzada.

Ademés, depenent de la base de dades, crearem uns usuaris amb privilegis diferents. Per finalitzar, realitzarem queries sobre la BBDD desnormalizada local.

OLAP and OLTP concepts explained.

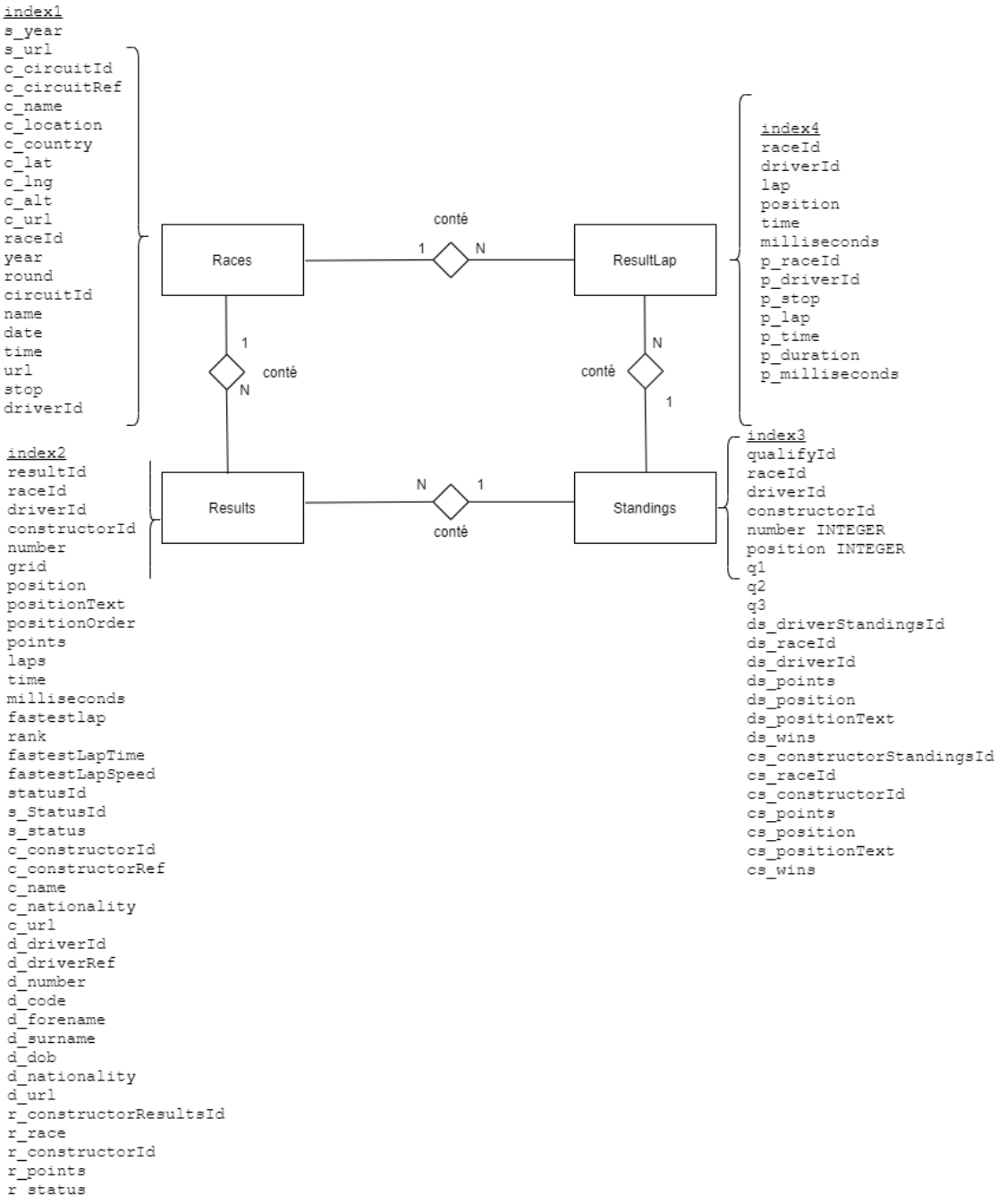
Una OLTP és una Base de Dades orientada a transaccions i està optimitzat per realitzar operacions de lectura i escriptura mentre que el format OLAP està pensat per realitzar funcions analítiques, l'informació d'aquestes es sol alimentar d'altres bases de dades ja plenes (com en el cas d'aquesta pràctica).

En el nostre cas, la OLTP és la base normalitzada mentre que la base de dades en format OLAP la fem servir per realitzar les quèries ja que, al ser un model de dades no normalitzada, aquestes s'executen més ràpid (hi ha molts menys productes cartesians entre taules).

Explain how you find out the remote OLTP database structure, include screenshots and commands with it explanation attached.
Un cop ens vam connectar al servidor PuigPedrós utilitzant el usuari i la contrasenya indicada en el enunciat vam connectar-nos a la base de dades F! utilitzant la comanda de SQL "USE F1". Un cop estem connectats, vem mirar quantes taules formaven aquesta base de dades amb la comanda (SHOW TABLES FROM F1).

Després, per cada taula, vem fer servir la comanda SELECT * FROM nomTaula per tal de veure el nom i tipus de cada columna.

Relational OLAP model diagram.



Explain how your Java/C program works.

Al iniciar el programa, mitjançant connectors, ens connectem tant a la base de dades remota., amb la comanda: mysql -u lsmotor_user -p

Un cop connectats, fem un “USE F1” per seleccionar la base de dades corresponent al projecte i després fem un “SELECT * FROM” de la primera taula de la OLTP i, mitjançant un Prepared Statement inserim cada fila de la la OLTP remota a la OLTP local.

```
aux2=smtRemote.executeQuery( s: "SELECT * FROM races;");
while (aux2.next()){
    PreparedStatement pstmt = conn.prepareStatement( s: "INSERT INTO races(raceId,year,round,circuitId,name,date,time,url)"+" VALU
    pstmt.setInt( i: 1,aux2.getInt( s: "raceId"));
    pstmt.setInt( i: 2,aux2.getInt( s: "year"));
    pstmt.setInt( i: 3,aux2.getInt( s: "round"));
    pstmt.setInt( i: 4,aux2.getInt( s: "circuitId"));
    pstmt.setString( i: 5,aux2.getString( s: "name"));
    pstmt.setDate( i: 6,aux2.getDate( s: "date"));
    pstmt.setTime( i: 7,aux2.getTime( s: "time"));
    pstmt.setString( i: 8,aux2.getString( s: "url"));
    pstmt.executeUpdate();
    pstmt.close();
}
System.out.println("OMPLINT RACES...OK");
```

Després, mitjançant triggers programats en SQL, inserim la informació a la taula de la OLAP corresponent.

Aquest procés es repeteix per cada taula de la OLTP (hem fet tants statements com taules en la OLTP).

Explain how you created the users (Grants granted...).

Primer de tot eliminem tots els possibles usuaris que ja existeixen en la nostre BBDD. Un cop estem segurs de que no existeixen usuaris en els nostres bases de dades (tant en la OLAP com en la OLTP) ens disposem a crear-los amb els permisos demanats.

El primer usuari, el analític, te permís per fer selects, crear i veure vistes. És per això que a aquest usuari li fent “`GRANT SELECT, SHOW VIEW, CREATE VIEW ON F1_OLAP.*`”

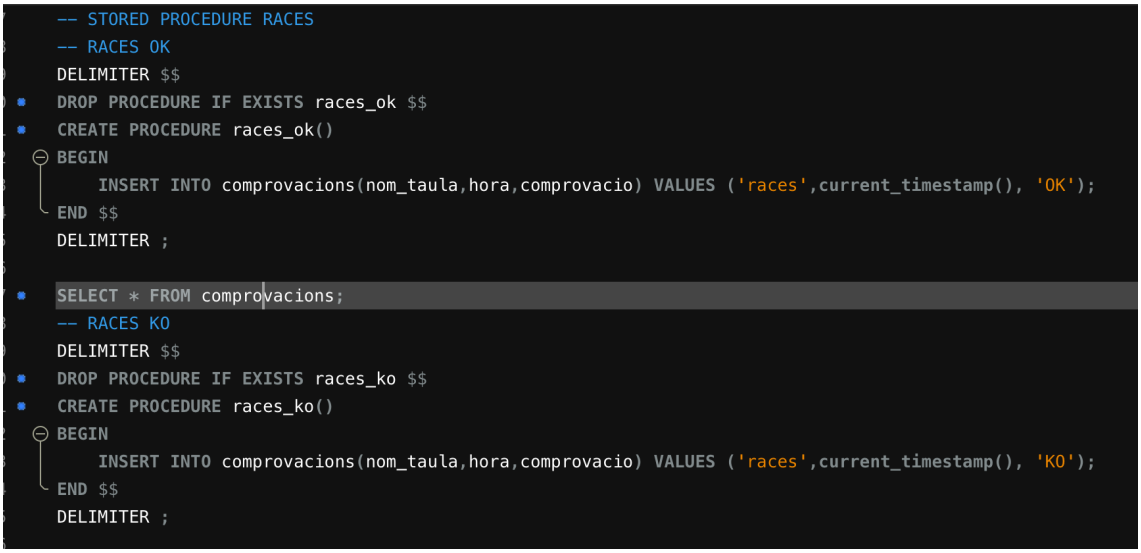
Al segon usuari, el manager, li donem permisos per tal que pugui fer updates tant en la OLAP com en la OLTP. És per això que li donem permís de updates tant en la nostre OLAP com en la nostre OLTP.

Per últim, l’usuari de relacions humanes, pot crear usuaris en qualsevol base de dades, per això aquest usuari té permís per crear usuaris en qualsevol base de dades “ `GRANT CREATE USER  ON *.*`”.

Explain how your stored procedures work.

Un cop hem importat correctament les dades de la OLTP a la OLAP hem de comprovar mitjançant un event si el nombre de files de la OLAP concorda amb el nombre de files de la OLTP que compleixen les condicions del Trigger.

Per comprovar si aquesta importació es correcta creem un event que conti el nombre de files, i depenent del resultat, mitjançant dos storeds procedures fem un insert a la taula on emmagatzema els status i inserim el nom de la taula que s'ha comprovat, l'hora de la comprovació i un estat que serà "OK" en cas que el nombre de files sigui correcte i "KO" si el nombre no concorda.



```
-- STORED PROCEDURE RACES
-- RACES OK
DELIMITER $$
DROP PROCEDURE IF EXISTS races_ok $$
CREATE PROCEDURE races_ok()
BEGIN
    INSERT INTO comprovacions(nom_taula,hora,comprovacio) VALUES ('races',current_timestamp(), 'OK');
END $$
DELIMITER ;

SELECT * FROM comprovacions;
-- RACES KO
DELIMITER $$
DROP PROCEDURE IF EXISTS races_ko $$
CREATE PROCEDURE races_ko()
BEGIN
    INSERT INTO comprovacions(nom_taula,hora,comprovacio) VALUES ('races',current_timestamp(), 'KO');
END $$
DELIMITER ;
```

Explain how your events and triggers works.

Triggers

A l'inici del script, fem el insert de les taules principals de la OLTP que formen les taules de la OLAP. Insertem la taula de Results, Races, Standings i ResultLap. Els valors que inserim en aquestes taules ens serviran més endavant per comprovar si l'informació d'altres taules de la OLTP s'han inserit ja, que no existeix en les taules anteriorment esmentades, o si simplement hem de modificar aquestes files, per tal, de completar l'informació.

```
-- Primer fem els inserts de les tres taules principals:
-- INSERt DE RESULTS
USE F1;
DELIMITER $$
DROP TRIGGER IF EXISTS F1.trigger_results $$
CREATE TRIGGER F1.trigger_results AFTER INSERT ON F1.results
FOR EACH ROW
BEGIN
    insert into F1_OLAP.RESUlTS(resultId, raceId, driverId, constructorId, number,grid, position, positio
    values (NEW.resultId, NEW.raceId, NEW.driverId, NEW.constructorId, NEW.number, NEW.grid, NEW.position

END $$
DELIMITER ;
SELECT * FROM F1_OLAP.results;
```

Un cop hem fet els inserts, passem a fer updates de les files no omplertes amb el insert comentat anteriorment.

Comencem per el uptades de la taula Races de la OLAP. Dins del trigger tenim un if. Si el year que s'acaba d'inserir en la taula seasons de la OLTP no existeix en la OLAP de races insertem aquest camp amb un insert. Si d'altre forma si que existeix en la OLAP, simplement fem un update em aquella fila i canviem el seu valor 0 (valor per defecte que hem fixat als INT que estan buits) per el nou any que hem inserit.

```
DELIMITER $$
DROP TRIGGER IF EXISTS F1.trigger_seasons$$
CREATE TRIGGER F1.trigger_seasons AFTER INSERT ON F1.seasons
FOR EACH ROW
BEGIN
    IF (SELECT COUNT(*) FROM F1_OLAP.races
    WHERE year = NEW.year ) = 0 THEN
        insert into F1_OLAP.races(s_url, s_year)
        values (New.url, New.year);

    ELSE
        UPDATE F1_OLAP.races
        SET s_url = NEW.url, s_year = NEW.year
        WHERE  F1_OLAP.races.year = NEW.year;
    END IF;
END $$
DELIMITER ;
```

Aquest mateix procediment el fem amb la taula Circuits de la OLTP. En aquest cas, mirem si el id del status que s'acaba d'inserir existeix en la OLAP de races. Com abans, si no existeix, inserim el id. Però, si existeix, només cal fer un update.

Després d'inserir informació en la taula de Status de la OLTP, comprovem si el id de driver que hem inserit existeix en la OLAP. Si no existeix, inserim la fila corresponen al id i, si existeix fem el update de la fila en la OLAP amb la fila que hem inserit a la OLTP.

```
DELIMITER $$
DROP TRIGGER IF EXISTS F1.trigger_status$$
CREATE TRIGGER F1.trigger_status AFTER INSERT ON F1.status
FOR EACH ROW
BEGIN
    IF (SELECT COUNT(*) FROM F1_OLAP.results
    WHERE StatusId = NEW.StatusId ) = 0 THEN
        insert into F1_OLAP.results(s_StatusId, s_status)
        values (NEW.StatusId,NEW.status);

    ELSE
        UPDATE F1_OLAP.results
        SET s_status = NEW.status, s_StatusId = NEW.statusId
        WHERE  F1_OLAP.results.statusId = New.statusId;
    END IF;

END $$
DELIMITER ;
```

Tot seguit anem fent els triggers de totes les taules de la mateixa manera, pero canviant el producte cartesià de les condicions per assegurar-nos que concorda amb la taula que hem fet el trigger de l'insert.

Events

La finalitat dels events és comprovar si s'ha importat l'informació de la OLTP a la OLAP de forma correcte.

Per fer això, creem una taula anomenada comprovacions que enmagatzemarà el nom de cada taula de la OLAP, l'hora a la que s'ha fet la comprovació i el estat de la comprovació. Aquest estat és una cadena que tindrà com a valor "OK" si el nombre de files és correcte i "KO" si el nombre és incorrecte.

```
USE F1;
DROP TABLE IF EXISTS comprovacions;
CREATE TABLE comprovacions(
    nom_taula VARCHAR(255),
    hora TIME,
    comprovacio VARCHAR(255)
);
SELECT * FROM comprovacions;
```

Per tal de que el nombre de files sigui correcte, cal que el nombre de files de la OLAP sigui igual al nombre de files de la OLAP que la formen.

Per tal de contar les files de la OLTP creem una taula anomenada comp, que contindrà el camp sum_igual. Aquest camp és un enter corresponent a cada taula de la OLTP que forma la OLAP. Si la suma d'aquests valors és igual al nombre total de files de la OLAP, l'importació s'haura fet de forma correcte.

```
DROP TABLE IF EXISTS comp;
CREATE TABLE comp(
    sum_igual INTEGER
);
SELECT * FROM comp;
```

La primera taula de la OLAP que comprovem és Races.

Comencem revisant quants id de circuits són iguals tant en la taula de circuits i races. Comprovem que hi han 1040 ids igual. Insertem aquest valor en la taula comp.

Després comprovem quants ids de circuits no apareixen en la taula races però si en circuits. Aquests que compleixen aquesta condició també els hem inserit a la taula comp amb un insert, en aquest cas només trobem una fila.

Fem el mateix amb el camp “year” de la taula races i la taula seasons. En aquest cas no hi ha cap valor que compleixi aquesta condició.

Després fem la suma de totes les excepcions en les que els id’s no concordan amb races i dels que si.

L’event de Races s’executa un cop al día. Controla si el nombre total de files de la OLAP de Races es igual a la suma dels valors de la taula comp.
Si aquesta condició es compleix, el event crida a un Stored Procedure que inserta “Races”, l’hora en que s’ha fet aquesta comprovació i, per últim, “OK” indican que el nombre de files és correcte. Si no es compleix la condició, l’event crida a un Stored Procedure que inserta la mateixa informació pero inserta “KO” en comptes de “OK”.

```
-- EVENTS RACES
DELIMITER $$
DROP EVENT IF EXISTS comprova_RACES $$
CREATE EVENT IF NOT EXISTS comprova_RACES
 ON SCHEDULE EVERY 5 MINUTE
COMMENT 'Comprovem la importacio de F1_OLAP.RACES'
DO BEGIN
    DECLARE equals INT DEFAULT 0;
    IF (SELECT COUNT(*) FROM F1_OLAP.RACES) = (SELECT SUM(sum_igual) FROM comp) THEN
            CALL races_ok();
        ELSE
            CALL races_ko();
    end if;
END $$
DELIMITER ;
```

Un cop comprovada la informació de Races, repetim el mateix proces per cada taula de la OLAP i fem les mateixes comprovacions en el moment de juntar la oltp.

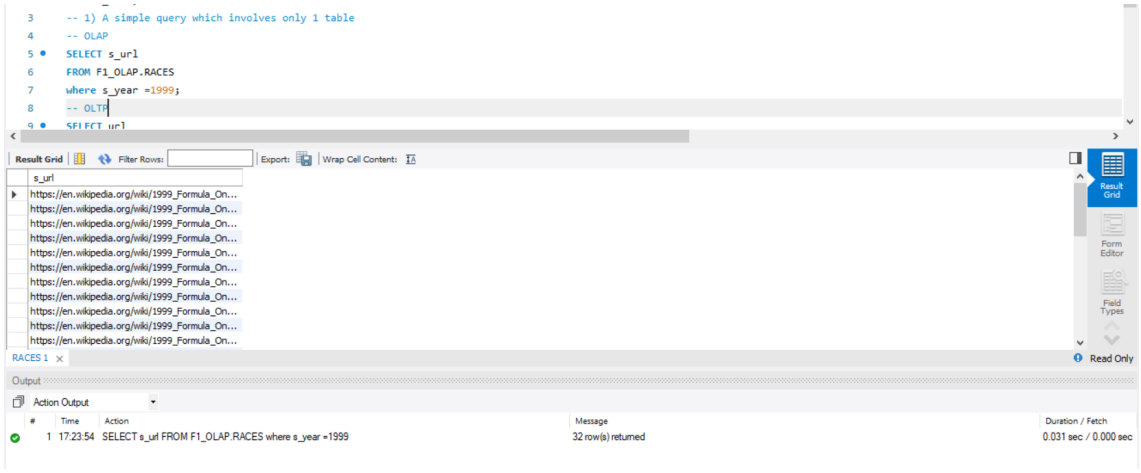
Tal com es pot veure, tots els events funcionan i demostren que la OLTP concorda amb la OLAP.

	nom_taula	hora	comprovacio
▶	races	21:00:34	OK
	standings	21:00:35	OK
	results	21:00:35	OK
	resultlap	21:00:37	OK

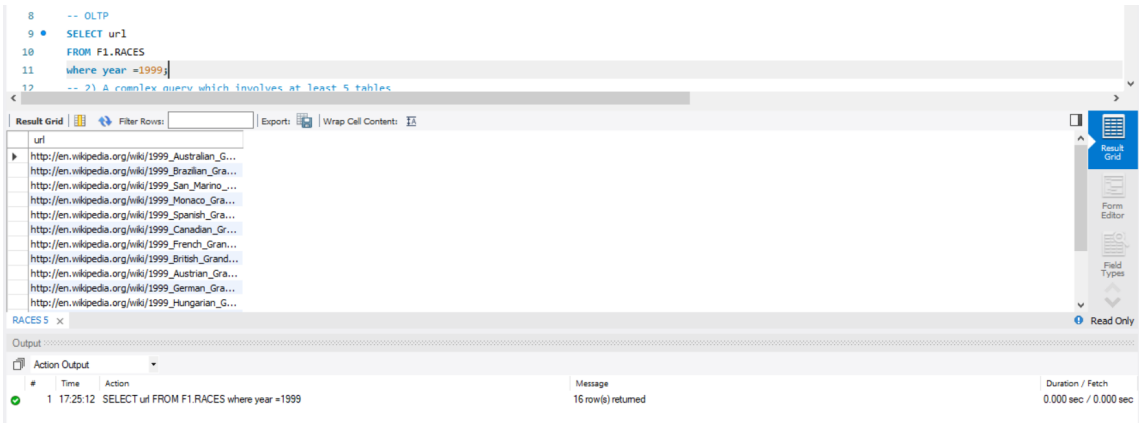
Report about OLTP vs OLAP comparison.
Mentres que en la OLTP tenim un model normalitzat, la OLAP esta composta per taules desnormalitzades. És per això que executem les sobre la OLAP, ja que s'executen més ràpid ja que hi han menys taula, per el que hem de fer menys productes cartesians.

Això ho hem vist a les queries on veiem que una mateixa sentència de SQL sobre la OLTP és molt més lenta que sobre la OLAP.

1. **A simple query which involves only 1 table**
   Aquesta query mostra el url de la temporada 1999.
   a. OLAP



   b. OLTP



2. **A complex query which involves at least 5 tables**

Aquesta query mostra el nom, cognom,nom de l'equip i la suma de tot el temps dels resultats, tenint en compte només les curses que han acabat.

a. OLTP

```
347    -- 2) A complex query which involves at least 5 tables
348    -- Aquesta query mostra el nom, cognom,nom del equip i tot el temps que ha fet pitstop de tots els c
349    -- OLTP
350
351    SELECT d.forename, d.surname, c.name,AVG(q.position) AS pos, ROUND(SUM(r.time)) AS time
352    FROM F1.Qualifying AS q, F1.Results As r, F1.Drivers AS d, F1.Status AS s, F1.Constructors AS c
353    WHERE  r.driverId = q.driverId AND d.driverId = q.driverId AND r.statusId = s.statusId
354    AND r.constructorId = c.constructorId
355    AND status LIKE '%Finished%' AND r.time IS NOT NULL
```

100% 1:359

Result Grid | Filter Rows: Search | Export: | Fetch rows:

forename	surname	name	pos	time
Ayrton	Senna	McLaren	1.0000	1727
Ayrton	Senna	Team Lotus	1.0000	1162
Ayrton	Senna	Toleman	1.0000	85
Lewis	Hamilton	McLaren	3.5640	311644
Lewis	Hamilton	Mercedes	3.5640	281088

Result 24

Action Output

	Time	Action	Response	Duration / Fetch Time
78	20:51:02	SELECT d.forename...	5 row(s) returned	5.862 sec / 0.000012...
79	20:53:32	SELECT d.forename...	5 row(s) returned	5.878 sec / 0.000010...

b. OLAP

Limit to 50000 rows

```
359
360    -- OLAP
361    SELECT r.d_forename, r.d_surname, r.c_name,AVG(s.position) AS pos,ROUND(SUM(r.time)) AS tim
362    FROM F1_OLAP.Results AS r,F1_OLAP.Standings AS s
363    WHERE r.driverId = s.driverId AND r.time IS NOT NULL AND r.s_status LIKE '%Finished%'
364    GROUP BY r.driverId, r.d_surname, r.d_forename, r.c_name
365    ORDER BY pos, r.c_name
366    LIMIT 5;
367
```

100% 9:366

Result Grid | Filter Rows: Search | Export: | Fetch rows:

d_forename	d_surname	c_name	pos	time
Ayrton	Senna	McLaren	1.0000	1727
Ayrton	Senna	Team Lotus	1.0000	1162
Ayrton	Senna	Toleman	1.0000	85
Lewis	Hamilton	McLaren	3.5640	311644
Lewis	Hamilton	Mercedes	3.5640	281088

Result 25

Action Output

	Time	Action	Response	Duration / Fetch Time
79	20:53:32	SELECT d.forename...	5 row(s) returned	5.878 sec / 0.000010...
80	20:54:08	SELECT r.d_forena...	5 row(s) returned	21.796 sec / 0.00001...

3. **An insert into 1 table**
   Aquesta query insereix a la taula de races l'any i el url de la temporada de formula 1 2021.
   a. **OLAP**

   b. **OLTP**

4. **An update into 1 field.**
   En aquesta query canviem l'any de totes aquelles files que continguin el url de la temporada 2021. Canviem l'any 2021 per 2022.

   a. **OLAP**

b. OLTP

```
35 •  UPDATE F1.SEASONS
36    SET year = 0
37    WHERE url = 'https://en.wikipedia.org/wiki/2021_Formula_One_World_Championship';
38    -- OLAP
39 •  UPDATE F1_OLAP.RACES
40    SET year = 0
41    WHERE url = 'https://en.wikipedia.org/wiki/2021_Formula_One_World_Championship';
42    -- 5) A delete into 1 table.
43    -- OLTP
44 •  DELETE FROM F1.SEASONS
45    WHERE year = 0;
46    -- OLAP
47 •  DELETE FROM F1_OLAP.RACES
48    WHERE year = 0;
49
```

```
Output
Action Output
#  Time      Action                                                                                                  Message                                                                  Duration / Fetch
1  17:34:01  UPDATE F1.SEASONS SET year = 0 WHERE url = 'https://en.wikipedia.org/wiki/2021_Formula_One_World_Ch...  1 row(s) affected Rows matched: 1  Changed: 1  Warnings: 0  0.000 sec
```

5. A delete into 1 table.

Aquesta query fa un delete de totes aquelles files que siguin del any 2022 i tinguin el url anterior.

a. OLAP

```
47 •  DELETE FROM F1_OLAP.RACES
48    WHERE year = 0;
49
```

```
Output
Action Output
#  Time      Action                                  Message                Duration / Fetch
1  17:49:27  DELETE FROM F1_OLAP.RACES WHERE year = 0  0 row(s) affected  0.000 sec
```

b. OLTP

```
44 •  DELETE FROM F1.SEASONS
45    WHERE year = 0;
46    -- OLAP
47 •  DELETE FROM F1_OLAP.RACES
```

```
Output
Action Output
#  Time      Action                                                                                                  Message                                                                  Duration / Fetch
1  17:34:01  UPDATE F1.SEASONS SET year = 0 WHERE url = 'https://en.wikipedia.org/wiki/2021_Formula_One_World_Ch...  1 row(s) affected Rows matched: 1  Changed: 1  Warnings: 0  0.000 sec
2  17:42:22  SELECT d.forename, d.surname, c.name,AVG(q.position),SUM(lt.time) FROM F1.Drivers AS d, F1.Qualifying AS q,...  Error Code: 1317. Query execution was interrupted  8.844 sec
3  17:42:31  INTERRUPT                                                                                               OK - Query cancelled                                                     0.000 sec
4  17:42:47  DELETE FROM F1.SEASONS WHERE year = 0                                                                   1 row(s) affected                                                        0.000 sec
```

Conclusions OLTP vs OLAP:

Després de moltes comprovacions i realitzar moltes queries de diferenta complexitat, és a dir, crear una query amb molts productes cartesians, un altre amb l'eina insert, un altre amb un update i per ultim una amb un delete.

Hem vist que el sistema OLAP és més ràpid per executar queries complexes com la comentada anteriorment. Queries que, si es fes servir un model OLTP seríen moltes més taules. El sistema OLAP és un sistema desnormalitzat fet que ens ajuda ha estalbiar-nos molts productes cartesians.

També hem vist com el sistema OLTP és molt més ràpid per fer petits inserts o updates ja que són taules més petites i triga menys temps en completar el procés.

Queries

1. **Find the statuses that never happened in the F1 history. Do this without using subqueries.**
   A l'inici de la inserció d'informació a la OLAP, insertem dos status diferents: uns son una copia de la taula status mentre que l'altre es l'informació de la taula status comparada amb aquelles taules de la OLAP.
   Al tenir dues taules de status diferents podem comparar-les entre si per obtenir els status que no han passat mai.

2. **Find the nationality and average time of those drivers whose teams have the lowest average pit stop time.**
   Igualem el id de constructor (que serà el id del equip) de la taula results amb el id de constructor que obtenimamb una subquery que té com a finalitat obtenir els equips amb menor temps de parada. Per saber això els ordenem en funció del camp milliseconds.

3. **Search for the drivers (complete name) who have beaten his own qualifying time for each successive round, that is, improving his time in each qualifying round, and that his fastest lap time on the race was faster than any of the qualifying time of any other driver for that race. Also, check that this achievement landed him in the podium after the race (any of the top three positions)**
   En aquesta query ens hem assegurat que sempre hagi quedat en el podi (la posició només pot ser 1,2 o 3), que hagi millorat el seu resultat en cada ronda i que la volta més ràpida estigui dins de totes aquelles voltes on no sigui el mateix conductor.

4. **Search for the drivers (complete name), the fastest speed and lap time and circuit name where the drivers have recorded the fastest lap in the race, but not the highest speed or the other way around. Tip: Don't check the fastest time with the column fastestLapTime in results table, use another information.**
   En aquesta query mirem que el id de conductors es trobi en aquell llistat de conductors amb la volta més ràpida però no en el llistar amb la més ràpida i viceversa.

5. **Check the biggest overtaking (specifying the driver's complete name, circuit, year and overtaking positions) in the F1 history, for the whole race (do not take into account the first lap), and in a lap period.**
   Primer de tot calculem la primera posició, es a dir la posició dels corredors en la segona volta, ja que, no tenim en compte la primera volta. Després calculem la posició dels corredors a la ultima volta, i tot seguit fem la resta de la primera posició i la ultima posició. A demes

calculem la posició de cada corredor i li restem la posició que tenia en la volta anterior per trobar els corredor que va més ràpids per volta.

Time dedicated per part.

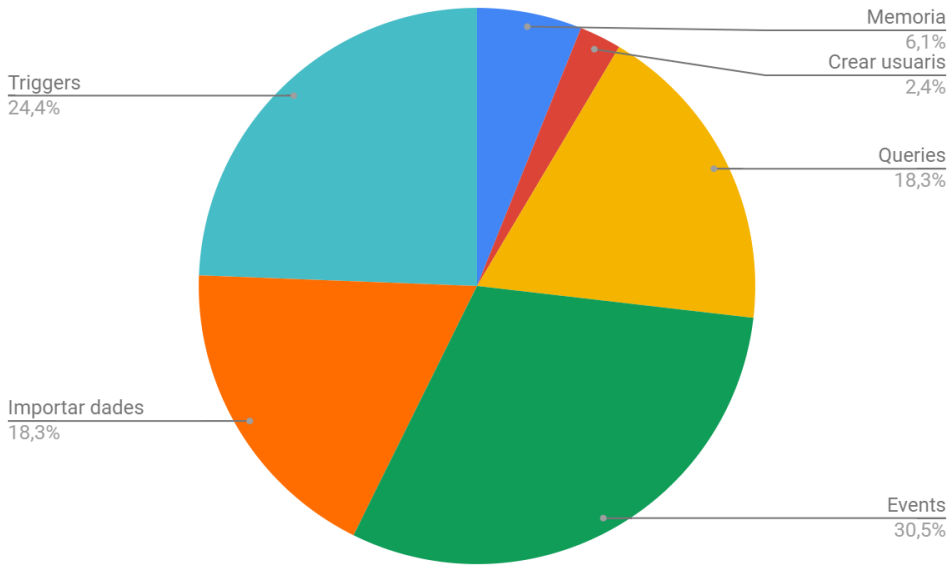
En un primer moment no sabíem com carregar la informació de la base de dades del servidor a la base de dades local. Finalment, després de buscar molt i provar moles coses diferents ho vem conseguir.

Un cop la OLTP local ja estava carregada calia buidar-la i tornar-la a omplir a mesura que ompliem la OLAP. Vem tardar molt per fer això ja que en un primer moment no sabiem que calia fer productes cartesians, creiem que només calia fer inserts.

Un cop vem trobar la forma de fer un trigger amb condicionals, vem passar a comprovar quela informació que havíem carregat fos la correcta. Això ens va portar molt de temps ja que no estavem molt familiaritzats amb els events i no sabíem com fer per calcuar el nombre de files que composa una OLAP en les diverses taules de la OLTP.

Per finalitzar, les queries ens van costar molt ja que vem trigar bastant a entendreles i conseguir mostrar el que creiem que és correcte.

Memoria	5
Crear usuaris	2
Queries	15
Events	25
Importar dades	15
Triggers	20



Conclusions

Aquesta pràctica ens ha ajudat molt per acabar de consolidar els coneixements donats a classe i ens ha permès tractar en un entorn molt proper a com seria en un treball profesional.

Durant la pràctica hem vist que hi han moltes formes de traspassar la informació d'una base de dades a una altre. En la pràctica hem utilitzat Prepared Staments, que són el millor mètode per calcar la base de dades (com vam fer amb la OLTP local amb la Remota). També hem fet servir Triggers per tal de passar la informació de la OLTP local. Aquest mètode és més farragós perquè, a diferència dels Statments, per a que aquests passin l'informaació d'una taula a una altre correctament, cal crear condicions.

També hem practicat els Events, mètodes molt eficients per tal de fer comprovacions sobre taules d'una o múltiples bases de dades. Aquests, com hem vist en la pràctica, es poden complementar amb altres mètodes, com els Stored Procedures o, fins i tot, Triggers.

Així doncs després de acabar la pràctica i comprovar la gran quantitat d'elements i mètodes que hem fet servir durant aquesta hem descobert el veritable potencial d'aquesta assignatura.

References as ISO 690:2010.

- Prueba de concepto de la conexión a SQL mediante Java. Disponible en:https://docs.microsoft.com/es-es/sql/connect/jdbc/step-3-proof-of-concept-connecting-to-sql-using-java?view=sql-server-ver15
- Curso Java desde cero.- Clase PreparedStatement (Bases de datos). Disponible en:https://www.youtube.com/watch?v=bkSCiT0GzBc
- SQL Server A trigger to work on multiple row inserts. Disponible en: https://stackoverflow.com/questions/2178889/sql-server-a-trigger-to-work-on-multiple-row-inserts
- Event Syntax. Disponible en:https://dev.mysql.com/doc/refman/5.7/en/events-syntax.html