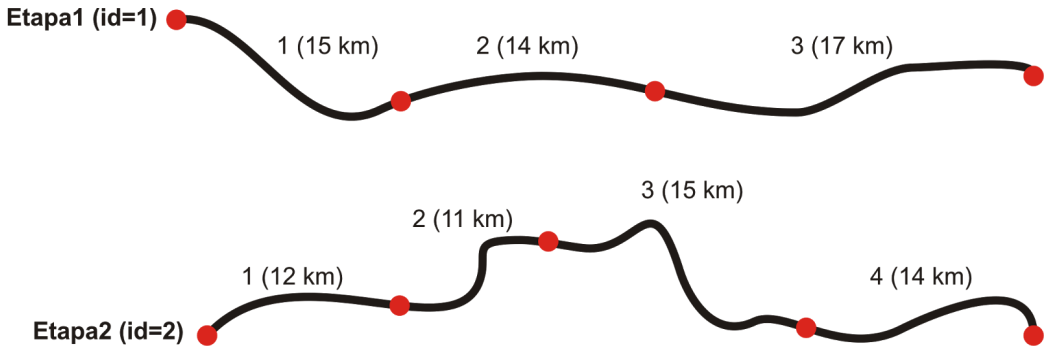
Baze podataka 1 **II kolokvijum (30 poena)** **23.12.2017**

Neka su date relacije koje čuvaju podatke o prolaznim vremenima vozaca na Dakar reliju:

tip_vozila (id_tipa, naziv) - spisak svih tipova vozila koja učestvuju na trkama (npr. motor, automobil, kamion...)
vozila (id_vozila, marka, idTipa) - spisak svih vozila koja učestvuju na reliju
vozaci (id_vozaca, ime, id_vozila) - spisak vozača i vozila sa kojim učestvuju na reliju
etape (idEtape, redniBrojStaze, duzinaStaze) - svaka etapa se sastoji iz nekoliko staza koje se voze odvojeno. Staze u etapi su date u vidu rednih brojeva. Na slici ispod su prikazane dve etape, dok su podaci koji ih opisuju dati u ekstanziji **etape.**
prolazna_vremena (idVozača, idEtape, redniBrojStaze, vremeStarta, vremeProlaskaKrozCilj) - spisak svih vožnji gde je za svaku dato vreme kada je vozač počeo da vozi i vreme kada je pro[ao kroz cilj. Ukoliko vozač nije završio trku, vrednost za **vremeProlaskaKrozCilj** će biti NULL. Jedan vozač može imati jedno prolazno vreme po stazi.

NAPOMENA: rad je potrebno predati u vidu **jednog** .sql fajla i pre svakog zadatka je potrebno naglasiti o kom zadatku se radi.



1. **(2 poena)** Napisati SQL upit kojim se određuje ukupna dužina svake etape.
2. **(3 poena)** Napisati SQL upit kojim se dobijaju imena vozača koji su prešli bar jednu stazu za manje od 1h.

ime
Alfan Pikar

3. **(2 poena)** Napisati SQL upit kojim se za svakog vozaca pronalaze staze koje nije vozio. U rezultatu se moraju naći imena vozača.
   *Marko Markovic nije vozio nijednu trku, ostali su vozili sve.
4. **(3 poena)** Napisati SQL upit kojim se dobijaju imena vozača koji su najmanje dve staze počeli da voze ali nisu prošli kroz cilj (ugnježdeni)

ime
Alan Kontre

5. **(5 poena)** Napisati SQL upit kojim se dobija apsolutni pobednik za svaki tip vozila. Apsolutni pobednik je onaj vozač koji ima najbolje prolazno vreme na svakoj stazi.

ime	naziv tipa
Alfan Pikar	motor
Latrik Perin	automobil

6. **(5 poena)** Napisati SQL upit kojim se za svaku stazu i svaki tip vozila pronalazi generalni plasman.

ime	id_tipa	id_etape	redni_broj_staze	plasman
Alfan Pikar	1	1	1	1
Alan Kontre	1	1	1	2
Sleser Mane	1	1	1	3
Mark Koma	1	1	1	4
Alfan Pikar	1	1	2	1
Alan Kontre	1	1	2	2
Sleser Mane	1	1	2	3
Mark Koma	1	1	2	4
Alfan Pikar	1	1	3	1
Sleser Mane	1	1	3	2
Mark Koma	1	1	3	3
Alfan Pikar	1	2	1	1
Alan Kontre	1	2	1	2
Sleser Mane	1	2	1	3
Mark Koma	1	2	1	4
Alfan Pikar	1	2	2	1
Sleser Mane	1	2	2	2
Mark Koma	1	2	2	3
Alfan Pikar	1	2	3	1
Alan Kontre	1	2	3	2
Sleser Mane	1	2	3	3
Mark Koma	1	2	3	4
Latrik Perin	2	1	1	1
AVatenen Ziro	2	1	1	2
Moro Moro	2	1	1	3
Kotulinski Lufelman	2	1	1	4
Latrik Perin	2	1	2	1
Moro Moro	2	1	2	2
AVatenen Ziro	2	1	2	3
Kotulinski Lufelman	2	1	2	4
Latrik Perin	2	1	3	1
AVatenen Ziro	2	1	3	2
Moro Moro	2	1	3	3
Kotulinski Lufelman	2	1	3	4
Latrik Perin	2	2	1	1
AVatenen Ziro	2	2	1	2
Moro Moro	2	2	1	3
Kotulinski Lufelman	2	2	1	4
Latrik Perin	2	2	2	1
AVatenen Ziro	2	2	2	2
Moro Moro	2	2	2	3
Kotulinski Lufelman	2	2	2	4
Latrik Perin	2	2	3	1
AVatenen Ziro	2	2	3	2
Kotulinski Lufelman	2	2	3	3
Moro Moro	2	2	3	4

7. Napisati SQL upit kojim se određuju vozači koji su bez padova u plasmanu, tj. oni koji nikada nisu imali lošiji plasman od nekog ostvarenog u prošlosti. **(5 poena)**
8. Napisati SQL upit kojim se dobija rang lista vozača automobila prema ukupnom broju bodovima koji se dobijaju na osnovu plasmana na stazama. Rezultujući upit treba da sadrži sledeće kolone **(5 poena)**:

   ime_vozaca, ukupan_broj_poena

poeni se ostvaruju na osnovu plasmana po stazama, prema sledecem bodovanju:

1. mesto 10 bodova
2. mesto 7 bodova
3. mesto 5 bodova
4. mesto 3 boda

ime_vozaca	ukupan_broj_poena
Latrik Perin	60
AVatenen Ziro	40
Moro Moro	30
Kotulinski Lufelman	20

```
/*  DML - Data Manipulation Language

        SELECT [{ALL | DISTINCT}] select_item [AS alias] [,...]
        FROM { table_name [[AS] alias] | view_name [[AS] alias]} [,...]
        [ [join_type] JOIN join_condition ]
        [WHERE search_condition] [ {AND | OR | NOT} search_condition [...] ]
        [GROUP BY group_by_expression{group_by_columns}
        [HAVING search_condition] ]
        [ORDER BY {order_expression [ASC | DESC]} [,...] ]

*/

/******** DATE FUNCTIONS *********

        DATEDIFF ( datepart , startdate , enddate )
        DATENAME ( datepart , date )
        DATEPART ( datepart , date )
        GETDATE  ( )
*/


/********* CONCAT && CAST *************

        SELECT CAST(column_name as TYPE)
        FROM table_name

        SELECT CONCAT(column_name, value, column_name, ...)
        FROM table_name
*/

/********* CASE ********

  I)
```

```
        Simple CASE expression:
        CASE input_expression
                WHEN when_expression THEN result_expression [ ...n ]
                [ ELSE else_result_expression ]
        END

  II)

        Searched CASE expression:
        CASE
                WHEN Boolean_expression THEN result_expression [ ...n ]
                [ ELSE else_result_expression ]
        END

*/
```