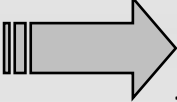


Eredü Erlazionala

TRIGGER-AK

(ABIARAZLEAK)

Trigger-ak / Abiarazleak

- Gertaeraren baten ondorioz DBKSk **inplizituki egikaritzen dituen prozedurak**, ekintza-segidak, dira  Datu-base Aktiboen oinarria
- Sybase SQL Server-etik sortuak
 - Ez dira SQL2-n (SQL92) azaltzen, **SQL3-n** ordekoa bai
- Adibidez,
 - Biltegian produktu baten 100 ale baino gutxiago baditugu, eskaera berria egin
 - Produktuaren iraungitze-data gaurkoa bada, arduradunari oharra bidali
 - 17:00etan DBren segurtasun-kopia egin

Triggerrak vs. Murriztapenak

- Integritate-murriztapenak zera adierazten dute:
 - Bete behar den baldintza (predikatua)
- Abiarazleek zera adierazten dute:
 - Bete behar den baldintza (murriztapenak bezala)
 - Zein egoeretan egiaztatu behar den baldintza hori
 - Baldintza betetzearen ondorioz zein ekintza eman behar diren aurrera

Trigerrak vs. Murritzapenak

- Produktu baten stock-ak ezin du 10etik jaitsi
 - **Integritate-murritzapena**
 - Gertaera (**inplizitua**): {update,insert} Produktu-ren gainean
 - Baldintza: check (stock > 10)
 - Ekintza (**inplizitua**): if condition = false then raise error
- Produktu baten stock-a 10etik jaisten denean, orduan produktuaren eskaera berri bat egin
 - **Negozio-politika**
 - Gertaera (**esplizitua**): Produktu-ren stock-aren update
 - Baldintza : check (stock < 10)
 - Ekintza (**esplizitua**): insert into Eskaera

Triggerren deskribapena

- Nola zehazten dira?
 - ezagutza-eredua
- Zein portaera dute exekuzio-denboran?
 - exekuzio-eredua

Trigger-aren osagaiak. Ezagutza-eredua

(ingelesez ECA eredua, Event-Condition-Action)

- **Gertaera:** trigger-a altxarazten duen jazoera
 - DB aldatzen duten INSERT, UPDATE, DELETE
 - Oracle-n gainera, CREATE, ALTER, LOGON, LOGOFF, STARTUP, SHUTDOWN, SERVERERROR, erabiltzaileek definitutakoak
- **Baldintza:** ekintza burutu behar denentz zehaztuko du
 - Adierazpen boolearra **TRUE ebaluatu** behar da
 - Baldintzarik jartzen ez bada, ekintza gertaera sortzen den bakoitzean burutuko da
- **Ekintza:** Trigger-a altxatzen denean **egikaritu beharrekoa**
 - Oracle: PL/SQL, Java, C prozeduretara deiak
 - SQL Server: Transact-SQL

Sintaxia SQL3n

```
CREATE TRIGGER <identifikatzailea>
{ AFTER | BEFORE | INSTEAD OF }
{ INSERT | DELETE | UPDATE [OF (<atrib> (, <atrib>)* ) ] }
  ON {<taula> | <bista>}
[ FOR EACH ROW | STATEMENT ]
[ REFERENCEING OLD AS <izena>  NEW AS <izena> ]
[ REFERENCEING OLD_TABLE AS <izena>  NEW_TABLE AS <izena> ]
[ WHEN <baldintza> ]
    <DBren gaineko ekintza-segida>
    <biltegitratutako prozedura bati egindako deia>
    <kanpoko funtzio bati egindako deia>
```

```
DROP TRIGGER <identifikatzailea>
```

```
REPLACE TRIGGER <identifikatzailea> <CREATE-ren aukera berdinak>
```

Adibidea

- Gertaera
 - Produktuen kopurua aldatzen da (*Produktua* taula)
- Baldintza
 - Biltegian geratzen dena ezarritako minimoa (100) baino txikiagoa da
- Ekintza
 - Produktuaren eskaera egin

```
CREATE TRIGGER EskaeraBerria
```

```
AFTER UPDATE OF kopurua ON Produktua G
```

```
FOR EACH ROW
```

```
WHEN NEW.kopurua < 100 B
```

```
EskaeraBerriaSortu ( ) E
```

←

```
INSERT INTO Eskaera VALUES (:OLD.kodea, 100+:OLD.kopurua, today())
```

...

Trigger-aren egoera

- **ENABLE** (aktiboa)
 - Altxarazi daiteke
 - Lehenetsitakoa (DBKSn egiaztatu)
- **DISABLE** (ez aktiboa)
 - Ezin da altxarazi

```
ALTER TRIGGER <identifikatzailea>  
{ENABLE | DISABLE}
```

Trigger-motak (I)

- Aurrekoa: **BEFORE** *trigger*
 - Gertaera sortu duen agindua egikaritu baino lehen burutzen da ekintza
 - tuplak sartu, ezabatu, aldatu edo ... baino lehen
- Ondorengokoa: **AFTER** *trigger*
 - Gertaera sortu duen agindua egikaritu ondoren burutzen da ekintza
 - tuplak sartu, ezabatu, aldatu edo ... ondoren
- Ordezkoa: **INSTEAD OF** *trigger*
 - Gertaera sortu duen aginduaren ordez egikaritzen da ekintza
 - tuplak sartu, ezabatu, aldatu edo ... ordez

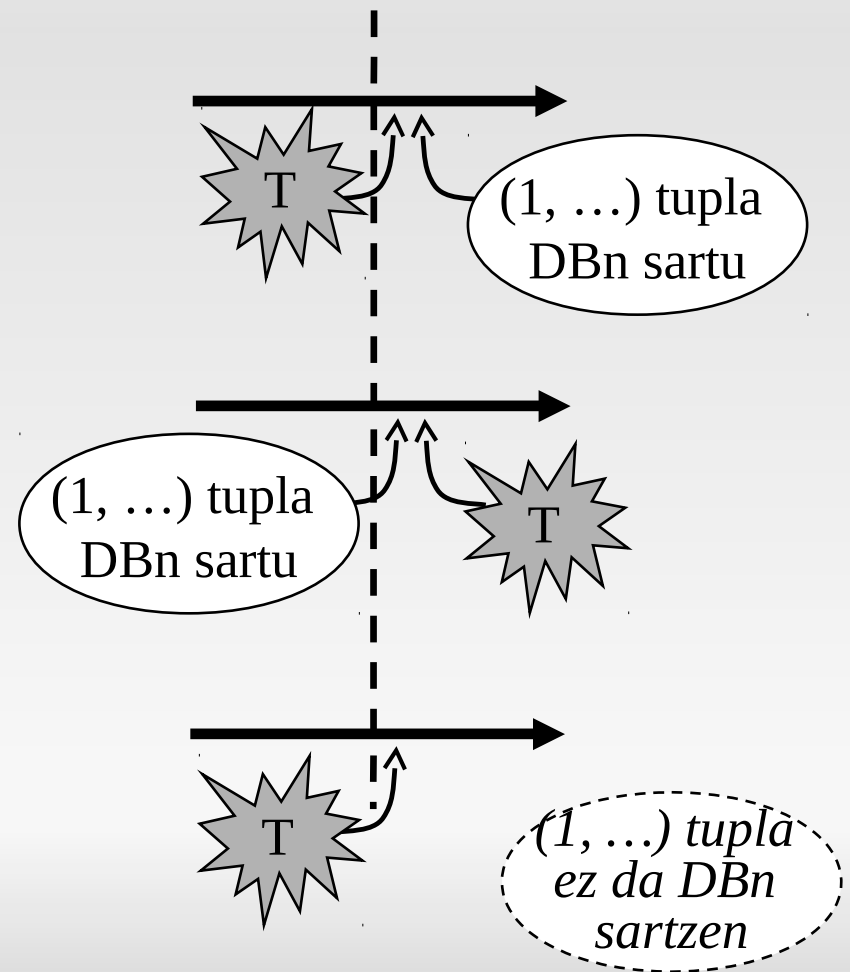
BEFORE-AFTER-INSTEAD OF. Alderaketa

```
CREATE TRIGGER T  
BEFORE INSERT  
ON Kontua  
mezuaBidali( )
```

```
CREATE TRIGGER T  
AFTER INSERT  
ON Kontua  
mezuaBidali( )
```

```
CREATE TRIGGER T  
INSTEAD OF INSERT  
ON Kontua  
mezuaBidali( )
```

INSERT INTO Kontua VALUES (1, ...)



Tupla mailakoa-Eragiketa mailakoa.

Alderaketa

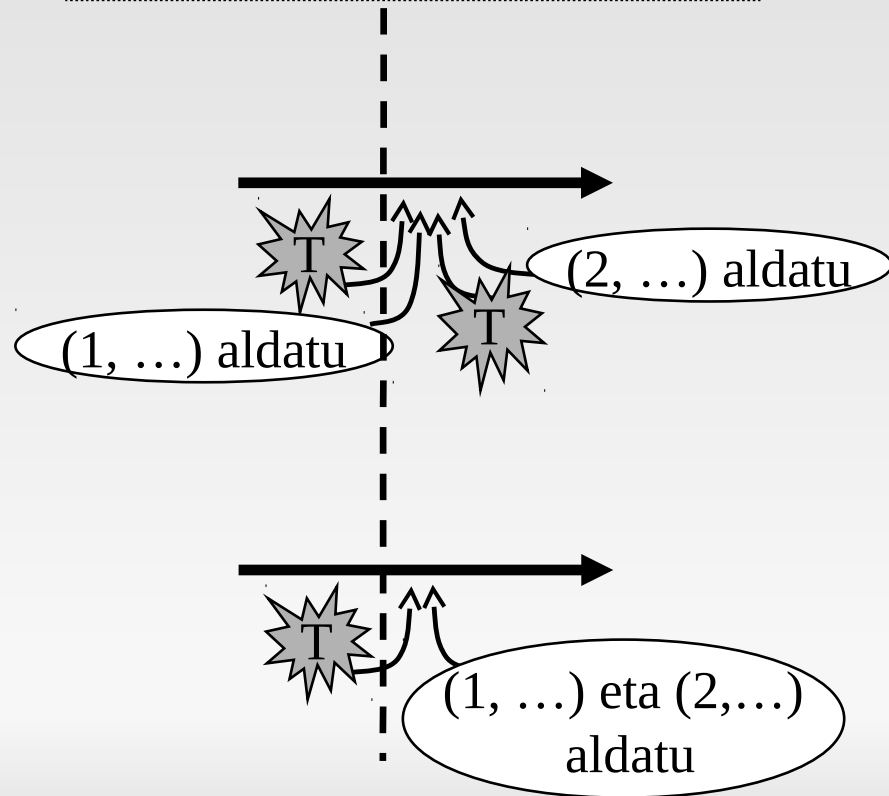
KONTUA

kkode	saldo	...
1	1.000	
2	500	
3	2.000	
...		

```
CREATE TRIGGER T
BEFORE UPDATE OF saldoa
ON Kontua
FOR EACH ROW
mezua( )
```

```
CREATE TRIGGER T
BEFORE UPDATE OF saldoa
ON Kontua
STATEMENT
mezua( )
```

```
UPDATE Kontua
SET saldoa = saldoa + 100
WHERE kkodea < 3
```



Trigger-motak (II)

- **Tupla mailakoa: *Row trigger* FOR EACH ROW**
 - Gertaera sortu duen agindua egikaritzean ukitzen duen tupla bakoitzarekin altxatzen da trigger-a
 - 5 tupla sartu \Rightarrow trigger-a 5 aldiz altxatzen da
 - tuplarik ez aldatu \Rightarrow trigger-a ez da altxatuko
 - DBn egindako aldaketak erreferentziazteko *NEW* eta *OLD* balioak erabil daitezke
- **Eragiketa mailakoa: *Statement trigger* STATEMENT**
 - Gertaera sortu duen agindua egikaritzean behin bakarrik altxatzen da trigger-a (zenbat tupla ukitzen dituen eta zein transakzioan kokatzen den kontuan hartu gabe)
 - 5 tupla sartu \Rightarrow trigger-a behin altxatzen da
 - tuplarik ez aldatu \Rightarrow trigger-a behin altxatzen da
 - DBn egindako aldaketak erreferentziazteko *NEW_TABLE* eta *OLD_TABLE* taulak erabil daitezke (SQL Server: *INSERTED*, *DELETED*)

Tupla mailakoak (*row trigger*)

- Ukitutako tupla bakoitzeko exekutatzen da

UPDATE Produktua
SET stock = stock -5
WHERE salneurria = 100

BALDIN ETA Xren stock-a aldatzean
10 unitatetik jaisten dela EGIAZTATZEN_BADA
ORDUAN X-ren eskaera berria egin

Kodea	Salneurria	...	Stock
1111	100		18
2222	100		13
3333	100		12
4444	100		12

4 aldiz pizten da
Ekintza 3 aldiz exekutatzen da



222-ren eskaera berria egin



333-ren eskaera berria egin



444-ren eskaera berria egin

Tupla mailakoak (*row trigger*)

Beste adibide bat

- Gertaera:
 - Produktu batzuren stock-unitateak aldatzen dira (Produktu taula)
- Baldintza:
 - Aldaketaren **ondoren** stock-eko unitate kopurua 10etik behera geratu da
- Ekintza:
 - Produktu honentzako eskaera berria egin

Gertaera

```
CREATE TRIGGER EskaeraBerria  
AFTER UPDATE OF Stock ON Produktu  
FOR EACH ROW
```

Baldintza

```
WHEN NEW.Stock < 10
```

Ekintza

```
BEGIN insert into Eskaera values(...) END;
```

Tupla mailakoak (*row trigger*)

- Abiarazlea ikututako tupla bakoitza aldatzen denean (ezabatu, txertatu, eguneratu) altxatzen da
 - 5 tupla ikutuak → 5 aldiz altxatzen da
 - 0 tupla ikutuak → ez da martxan ipintzen
- **OLD** eta **NEW** aldagaiak tupla dute aldaketaren aurretik (OLD) eta aldaketa gertatu ondoren (NEW)
 - Gertaera INSERT bat bada, OLD aldagaiak ez du zentzurik
 - Gertaera DELETE bat bada, NEW aldagaiak ez du zentzurik

Eragiketa mailakoak (Statement *trigger*)

- Taula osorako BEHIN exekutatzen da

```
UPDATE Produktua  
SET stock = stock -5  
WHERE Salneurria = 100
```

BALDIN ETA Xren stock-a aldatzean
Unitate kopuru totala 100etik jaisten dela EGIAZTATZEN BADA
ORDUAN 5 gradutan jaitsi aire girotua

Kodea	Salneurria	...	Stock
1111	100		18
2222	100		13
3333	100		12
4444	100		12



“5 gradutan jaitsi aire girotua”

Eragiketa mailakoak (Statement *trigger*)

- Gertaera:
 - Produktu batzuren unitateak aldatzen dira (Produktu taula)
- Baldintza:
 - Aldaketaren **ondoren** unitate kopuruaren TOTALA 100etik behera geratu da
- Ekintza:
 - 5 gradutan jaitsi aire girotua

```
CREATE TRIGGER EskaeraBerria  
AFTER UPDATE OF unitateak ON Produktua  
STATEMENT  
WHEN 100 > (SELECT SUM(unitateak) FROM NEW_TABLE)  
BEGIN setAireGirotua(-5) END;
```

Eragiketa mailakoak (Statement *trigger*)

- Abiarazlea BEHIN SOILIK altxatzen da (ikututako tuplak kontuan izan gabe)
 - 5 tupla ikutuak → behin altxatzen da
 - 0 tupla ikutuak → behin altxatzen da
- **OLD_TABLE** eta **NEW_TABLE** aldagaiek ikututako tuplen MULTZOA dute aldaketaren aurretik eta ondoren (SQL Server: INSERTED, DELETED)

Trigger-en definiziorako konbinaketak

- *BEFORE statement trigger*

- Gertaera sortu duen agindua egikaritu baino lehen burutzen da ekintza

- *BEFORE row trigger*

- Gertaera sortu duen aginduak ukitzen duen tupla bakoitza aldatu baino lehen eta dagozkion integritate-murriztapenak egiaztatu baino lehen burutzen da ekintza

- *AFTER row trigger*

- Gertaera sortu duen aginduak ukitzen duen tupla bakoitza aldatu ondoren eta erlazionatutako integritate-murriztapenak egiaztatu ondoren burutzen da ekintza uneko tuplarentzat (tupla blokeaturik geratzen da)

- *AFTER statement trigger*

- Gertaera sortu duen agindua egikaritu ondoren eta dagozkion taularteko integritate-murriztapenak (asertzioak) egiaztatu ondoren burutzen da ekintza

BEFORE eta AFTER vs. INSTEAD OF

- BEFORE eta AFTER

Oinarri-tauletarako bakarrik

EZ bistetarako

- Bista batean tuplaren bat sartzen/ezabatzen/aldatzen bada eta bere oinarri-taulek BEFORE edo AFTER motako trigger-ik definituta badute, orduan hori altxaraziko da

- INSTEAD OF

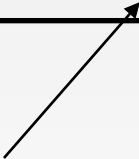
Bistetarako

- Aldagarriak ez diren bistak aldatzeko modu gardena da
- Bista batean egindako INSERT eragiketarekin erlazionatutakoa zein den jakinez gero, eragiketa hori trigger batean adieraz daiteke
- Trigger-mota hauekin Oraclen ezin da WHEN klausula jarri

Adibideen domeinuak

Bezerao(nan, izena, helbidea, soldata)
Kontua(kkodea, *nan*, saldoa, sorreradata)
Mailegua(mkodea, *kkodea*, zenbatekoa, data)

Langilea(nan, izena, soldata, *beresaila*)
Saila(sailzenb, sizena, aurrekontua, *zuzendaria*)



*Sailean atxikirik dauden
langileen soldataren batuketa*

Adibidea: BEFORE statement

Negozio-araua: Langileek asteburuetan ezingo dituzte kontu korronteei buruzko datuak aldatu

```
CREATE TRIGGER langileMaltzurrikEz  
BEFORE DELETE OR INSERT OR UPDATE OF (saldoa)  
ON Kontua  
WHEN  
    (DAYOFWEEK(today( ))='Sat' OR  
    DAYOFWEEK(today( ))='Sun')  
ZuzendariariOharra_prozedura( )
```

```
graph LR; Gertaera([Gertaera]) --> EventList[BEFORE DELETE OR INSERT OR UPDATE OF (saldoa)]; Baldintza([Baldintza]) --> Condition[(DAYOFWEEK(today( ))='Sat' OR DAYOFWEEK(today( ))='Sun')]; Ekintza([Ekintza]) --> Action[ZuzendariariOharra_prozedura( )];
```

Adibidea: BEFORE row

Negozio-araua: Mailegu berri bat eskatzean, eskaera egin duen bezeroaren soldata egiaztatuko da, maileguaren zenbatekoa ezingo da soldata baino 100 aldiz handiagoa izan

```
CREATE TRIGGER maileguBerria
```

```
BEFORE INSERT ON Mailegua
```

```
FOR EACH ROW
```

```
WHEN (NEW.zenbatekoa > 100 *
```

```
    (SELECT soldata
```

```
      FROM Bezeroa NATURAL JOIN Kontua
```

```
      WHERE kkodea = NEW.kkodea))
```

```
MaileguaAtzeraBota( )
```

Gertaera

Baldintza

Ekintza

Adibidea: BEFORE row

Negozio-araua: Saldo negatiboa duen kontu korrante berria sartzean edo kontu korrantea aldatuz gero, saldo negatiboarekin uztean, saldoan 0 jarri eta, edo mailegu berria sortu (zenbatekoa=zordun_saldoa duena), edo aurreko maileguari zordun-saldoa gehitu

```
CREATE TRIGGER zordunSaldoa
BEFORE INSERT OR UPDATE OF (saldoa) ON Kontua
FOR EACH ROW
WHEN (NEW.saldoa < 0)
BEGIN
  IF EXISTS (SELECT * FROM Mailegua WHERE kkodea=:NEW.kkodea)
  THEN
    UPDATE Mailegua SET zenbatekoa=zenbatekoa+:NEW.saldoa
    WHERE kkodea = :NEW.kkodea
  ELSE
    INSERT INTO Mailegua
    VALUES (autokode( ), :NEW.kkodea, :NEW.saldoa, today( ))
  END IF;
  NEW.saldoa = 0;
END
```

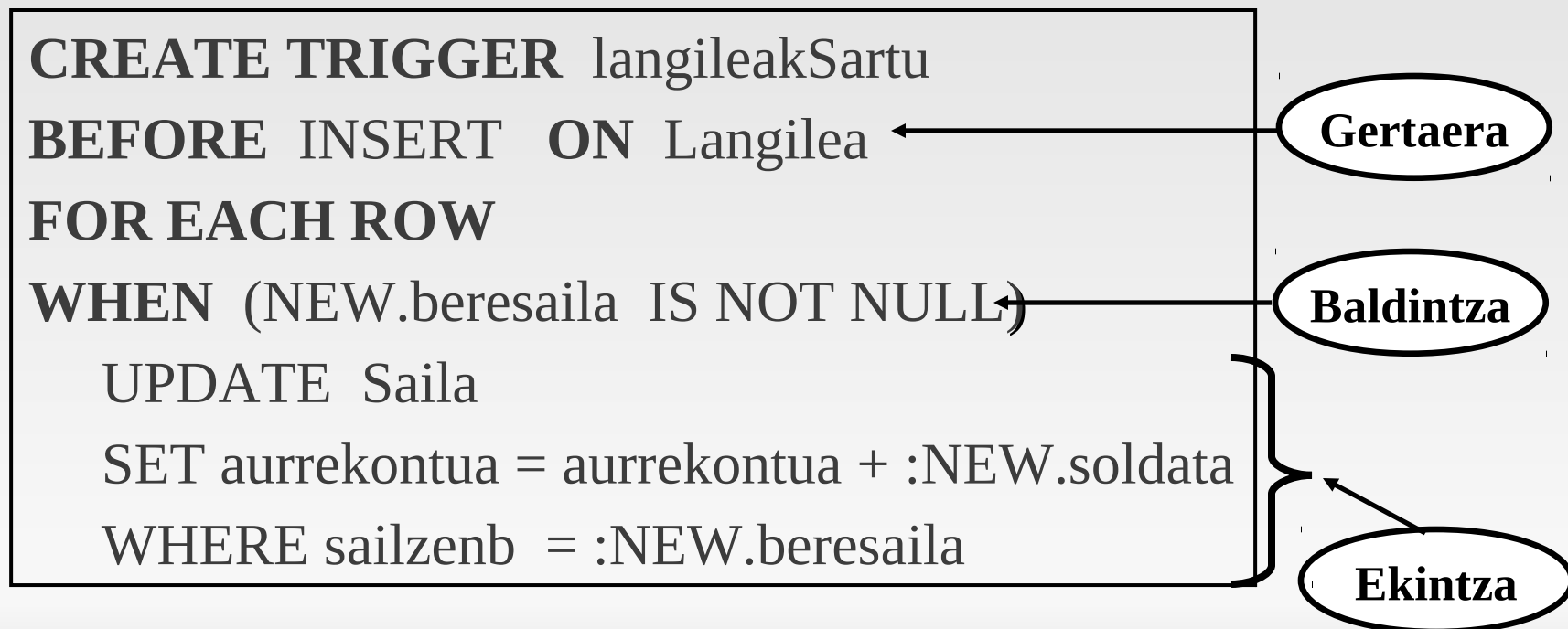
Gertaera

Baldintza

Ekintza

Adibidea: Datu eratorriak (1)

Negozio-araua: Langile berria sartzean, hau sailen batean atxikirik badago, sail horren aurrekontua behar bezala eguneratu beharko da



Adibidea: Datu eratorriak (2)

Negozio-araua: Langile baten (edo gehiagoren) soldata aldatzean, langilea sailen bati atxikirik badago, sailaren aurrekontua behar bezala eguneratu beharko da

```
CREATE TRIGGER langileenSoldataAldatu  
BEFORE UPDATE OF (soldata) ON Langilea  
FOR EACH ROW  
WHEN (NEW.beresaila IS NOT NULL)  
    UPDATE Saila  
    SET aurrekontua = aurrekontua + :NEW.soldata - :OLD.soldata  
    WHERE sailzenb = :NEW.beresaila
```

Gertaera

Baldintza

Ekintza

Adibidea: Datu eratorriak (3)

Negozio-araua: Langile baten (edo gehiagoren) saila aldatzean, aurreko eta ondorengoko sailen aurrekontuak eguneratu beharko dira

```
CREATE TRIGGER langileakSartu  
BEFORE UPDATE OF (bereSaila) ON Langilea  
FOR EACH ROW
```

```
    UPDATE Saila
```

```
    SET aurrekontua = aurrekontua + :NEW.soldata
```

```
    WHERE sailzenb = :NEW.beresaila;
```

```
    UPDATE Saila
```

```
    SET aurrekontua = aurrekontua - :NEW.soldata
```

```
    WHERE sailzenb = :OLD.beresaila;
```

Gertaera

~~Baldintza~~

Ekintza

Adibidea: Datu eratorriak (4)

Negozio-araua: Langilea kanporatzen denean sailen bati atxikirik badago, honen aurrekontua eguneratu beharko da

```
CREATE TRIGGER langileakBota
BEFORE DELETE ON Langilea
FOR EACH ROW
WHEN (OLD.beresaila IS NOT NULL)
    UPDATE Saila
    SET aurrekontua = aurrekontua - :OLD.soldata
    WHERE sailzenb = :OLD.beresaila;
```

Gertaera

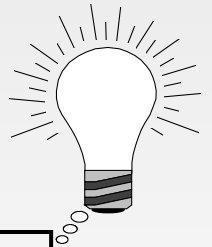
Baldintza

Ekintza

Adibidea: **INSTEAD OF (I)**

```
CREATE VIEW SailaDutenLangileak AS  
SELECT Langilea.izena, Saila.sizena  
FROM Langilea INNER JOIN Saila ON beresaila = sailzenb
```

Bista ez da aldagarria !!!
Join bat du, eta
taulen gakoak ez dira azaltzen



DBK-k bistaren gainean egindako aldaketak nola ulertu behar diren kontuan har dezake, eta *INSTEAD OF trigger* baten moduan inplementa dezake

LANGILEA

na	izena	bereSaila
----	-------	-----------

1	Jon	S1
2	Miren	S2
3	Aitor	S2

UPDATE SDL
SET sizena = 'KAT'
WHERE izena = 'Jon'

LANGILEA

na	izena	bereSaila
1	Jon	S2
2	Miren	S2
3	Aitor	S2

UPDATE LANGILEA

✓ **SET bereSaila = 'S2'**
WHERE izena = 'Jon'

SAILA

sailzenb	sizena
----------	--------

S1	LSI
S2	KAT

SDL

izena	sizena
Jon	KAT
Miren	KAT
Aitor	KAT



SDL izena sizena

Jon	LSI
Miren	KAT
Aitor	KAT

Zein
oinarri-taula
aldata?

SAILA

sailzenb	sizena
S1	KAT
S2	KAT

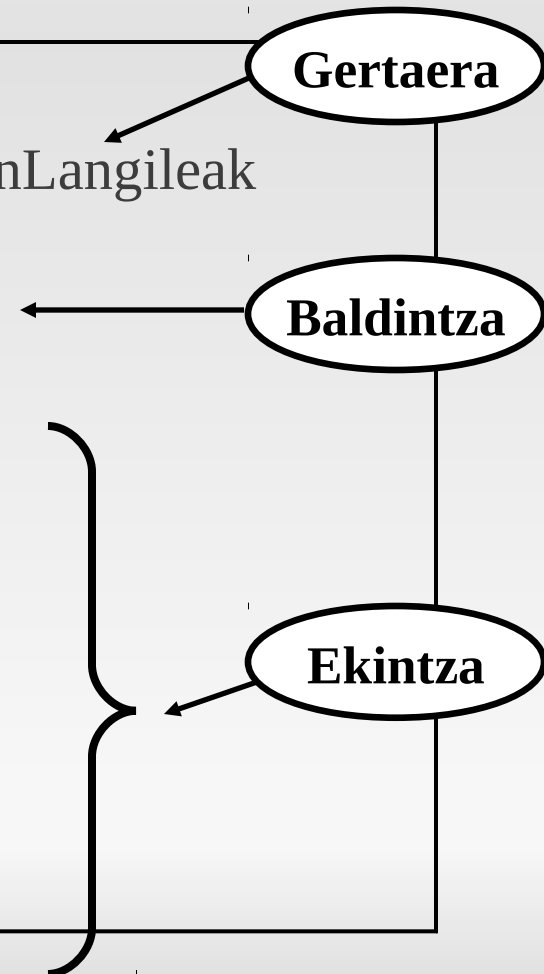
UPDATE SAILA

SET sizena = 'KAT'
WHERE sailzenb = 'S1'

Adibidea: INSTEAD OF (II)

Bista honekin aurretik existitzen diren sailetara atxeki daitezke langileak

```
CREATE TRIGGER bistaEguneratu  
INSTEAD OF UPDATE OF (sizena) ON SailaDutenLangileak  
WHEN EXISTS (SELECT * FROM Saila  
                WHERE sizena = NEW.sizena)  
DECLARE sailarenZkia NUMBER  
BEGIN  
    SELECT sailzenb INTO :sailarenZkia  
    FROM Saila WHERE sizena = :NEW.sizena;  
    UPDATE Langilea SET bereSaila = :sailarenZkia  
    WHERE izena = :NEW.izena  
END
```



Egikaritzapenaren algoritmoa

Datu-basea aldatuko duen **E eragiketa** egikaritzeko unean:

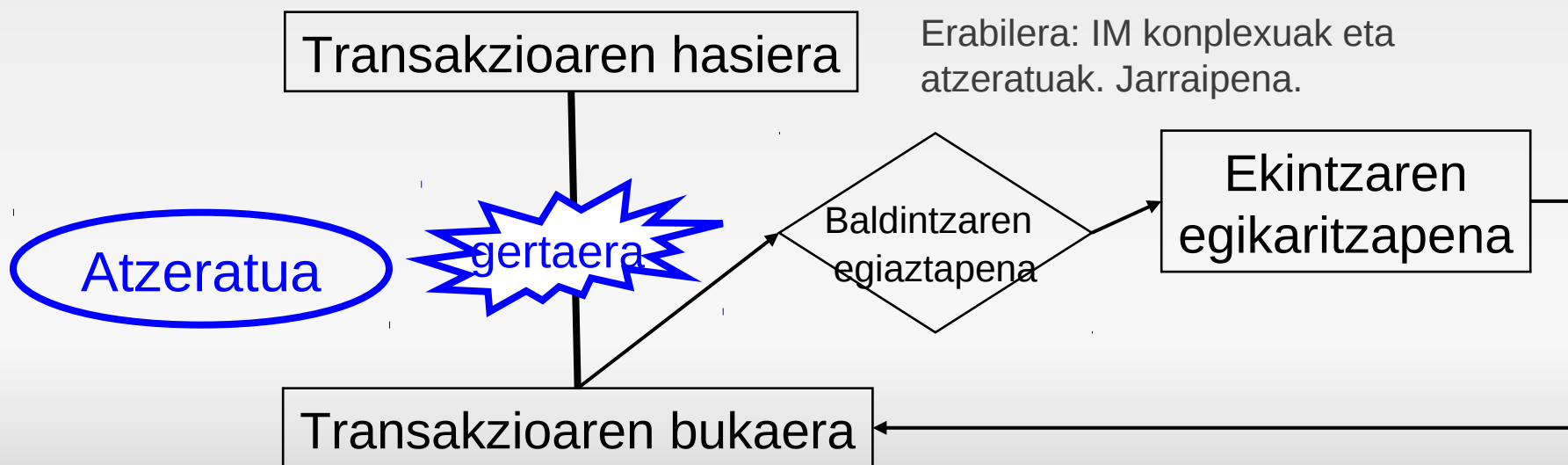
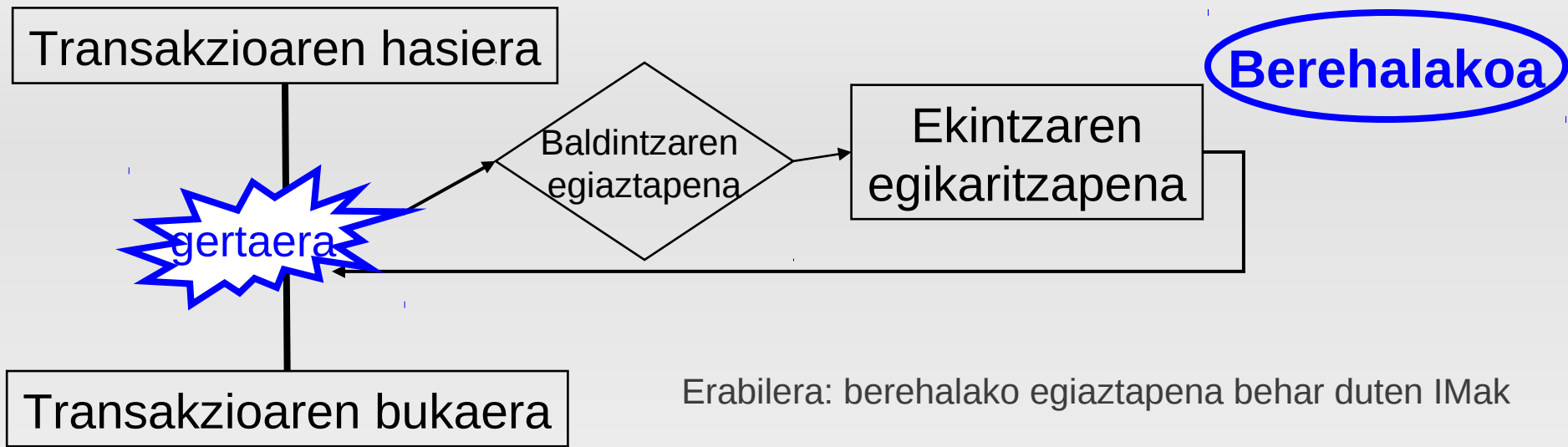
1. E-ri lotutako **BEFORE statement** motako trigger-ak egikaritu
2. Eragiketak ukituko duen **tupla bakoitzeko**
 - Bere **BEFORE row** motako trigger-ak egikaritu
 - Tupla blokeatu, bere gainerako **E eragiketa burutu** eta balizko **integritate-murriztapenak egiaztatu** (blokeoa ez da transakzioa hitzartu arte askatuko)
 - Bere **AFTER row** motako trigger-ak egikaritu
3. Taularteko integritate-murriztapenak egiaztatu
4. E-ri lotutako **AFTER statement** motako trigger-ak egikaritu

Trigger-a altxarazteko unea.

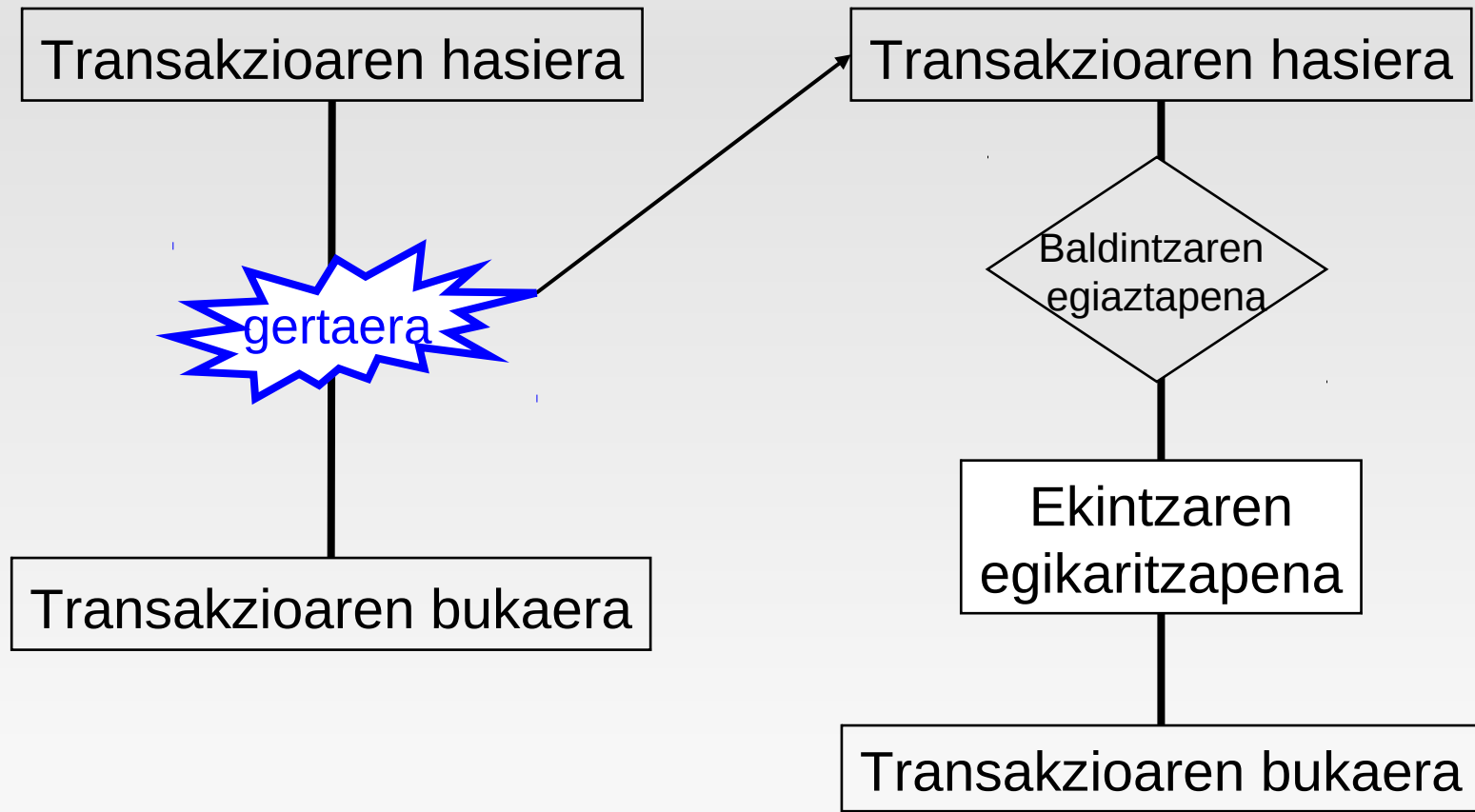
Exekuzio-eredua

- *Berehalakoa (IMMEDIATE)*: Gertaera jazo bezain laster, trigger-a altxarazi duen aginduaren transakzio berean
 - Baldintza, trigger-a altxarazi duen aginduaren aurretik, atzetik edo ordeztatuko da (BEFORE, AFTER, INSTEAD OF)
- *Atzeratua (DEFERRED)*: Trigger-a altxarazi duen agindua dagoen transakzioaren bukaeran
- *Banatua (DETACHED)*: Baldintza beste transakzio batean egiaztatzen da, trigger-a altxarazi duen transakziotik banatuta

Trigger-a altxarazteko unea: Berehalakoa vs. Atzeratua



Trigger-a altxarazteko unea: Banatuta



Erabilera: IM konplexuak eta atzeratuak.

Trigger-en erabilpenak (I)

- Datuen sendotasuna kontserbatzea
 - Baliozkoak ez diren transakzioak ekidin
 - Bestelako integritate-murriztapenak
 - Datu-base Banatuetan dauden taulen arteko integritate erreferentziala zaintzeko murriztapenak
 - Bide arruntetatik adierazi ezin diren beste integritate erreferentzialak adierazteko
 - NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK
 - DELETE CASCADE, UPDATE CASCADE, DELETE SET NULL...
- Negozio-arau konplexuak betetzen direla bermatzea
- Eratorritako datuak mantentzea
 - Eratorritako zutabeak automatikoki eguneratzea
 - Biltegiaratuak bistak mantentzea
 - Bisten oinarri-taulen datuak aldatzea
 - Taulen errepikapen sinkronizatuak mantentzea

Trigger-en erabilpenak (II)

- Zenbait gertaeren auditoria egitea (DBKSk eskaintzen dituenez gain)
 - Interestzen zaizkigun gertaeren lorratz (“log”) fitxategia mantentzea
 - Taulen eguneraketei buruzko estatistikak jasotzea
- DBKSren kanpo dauden beste sistema batzuk deitzea
 - DBKSrekin erlazioa duten aplikazioak
 - DBren gertaerei buruzko informazioa bidaltzeko
- Erabiltzaileei datuen egoerari buruzko oharrak bidaltzea
 - mezu elektronikoa edo telef. mugikorrera mezua bidali

Abantailak

- DBKSk negozio-arauak zentralizatu eta aplikatzen ditu
 - aplikazioen konplexutasuna murrizten da

Eragozpenak

- Datu-baseak konplexuagoak dira
 - DBak diseinatu, inplementatu eta kudeatzea zailagoa da
 - Trigger-en logika sartu behar delako
 - Ezkutatutako funtzionalitatea
 - Eragiketa sinple batek albo-ondorioak sor ditzakeelako
 - Erabiltzaileak ez du DBn gertatutakoari buruzko kontrolik
 - Kontraesanean egon litezkeen trigger-ak egon daitezke
- Sistema lanez gainezka jar dezakegu
 - Sistemak trigger-en baldintzak egiaztatu behar ditu DBren gainerako eragiketa bakoitzeko
 - Eraginkortasuna txikiagoa: zenbat eta trigger gehiago, okerrago
- Ez da bukaera bermatzen
 - SQL Server eta Oracle-n gehienez ere 32 maila onartzen dira

Trigger-ak hainbat sistematan

	SQL3	Oracle	MSSQL	Informix	Ingres	Rdb	Interbase
before aukera	Bai	Bai	Bai	Bai	Bai	Bai	Bai
after aukera	Bai	Bai	Bai	Bai	Bai	Bai	Bai
instead_of aukera	Bai	Ez	Bai	Ez	Ez	Ez	Ez
baldintza	Bai	Bai	Bai	Bai	Bai	Bai	Ez
row triggers	Bai	Bai	Ez	Bai	Bai	Bai	Bai
old/new errefer.	Bai	Bai	Ez	Bai	Bai	Bai	Bai
statement triggers	Bai	Bai	Bai	Bai	Ez	Ez	Ez
old/new table errefer.	Bai	Ez	Bai	Ez	Ez	Ez	Ez
lehentasunak	Bai	Ez	Ez	Ez	Ez	Ez	Bai
jarraikako altxatzeak	Bai	Bai	Bai	Bai	Bai	Bai	Bai
jarraikakoen sakonera	infinitu	32	32	61	20	infinitu	?