ABIS NIEUWSBRIEF VOOR DB2 PROFESSIONALS

JAARGANG 1 - NUMMER 1, SEPTEMBER 2002

OPEN CURSOR

Welkom bij het eerste nummer van 'Exploring DB2'.
'Exploring DB2' wordt een technische publicatie, voor professionele DB2 gebruikers. Voor professionele DB2 gebruikers, omdat de besproken topics zullen verder bouwen op een aantal bekende, traditionele DB2 vaardigheden, juist vereist om met de steeds nieuwe features van DB2 aan de slag te gaan.
Een technische publicatie, aangezien de meeste artikelen aan de hand van scripts zullen worden geïllustreerd.
Meer specifiek zullen de behandelde topics behoren tot de volgende domeinen:
Nieuwe technieken in applicatieontwikkeling;
Performance en efficiëntie;
Beheer en organisatie.
U merkt het - echt iets om naar uit te kijken. Veel leesgenot!

Het ABIS DB2 team.

IN DIT NUMMER:

- *Het gebruik van optimizer hints in DB2* - of hoe u de DB2 optimizer eindelijk zelf kunt meedelen hoe moeilijke SQL moet worden geoptimizeerd.
- *Dossier 7*: artikelen omtrent DB2 versie 7 weetjes. In dit nummer behandelen we Web services.
- Opeenvolgende keys toekennen in DB2 - u gebruikt timestamps? CICS temporary queues? Eindelijk kan u ze ook door DB2 laten genereren. Lees *1, 2, 3, 4, 5, ... en wat daarna? Over identity kolommen (i).*
- *Cursusplanning sep-dec 2002*.

CLOSE CURSOR

In het volgende nummer besteden we voornamelijk aandacht aan de problematiek van de 'Pendant Actions' bij het coderen van business logica in triggers. En 'Dossier 7' staat stil bij de Catalog.

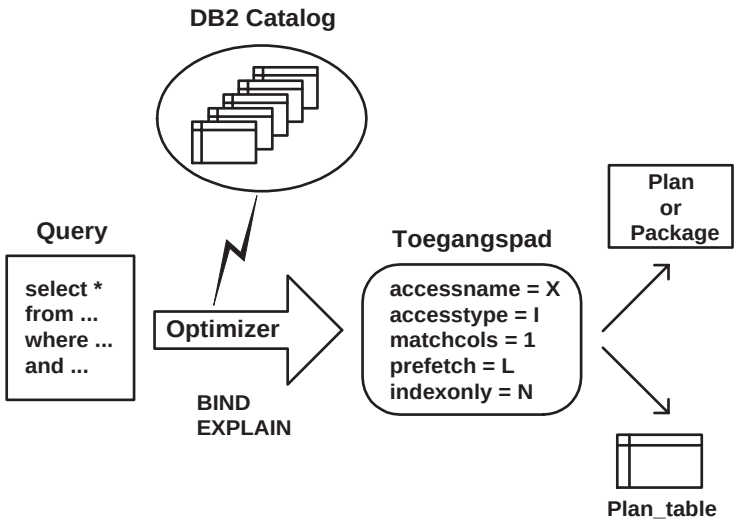
Tot dan!

Het gebruik van optimizer hints in DB2

Diane Hendrix (ABIS)

Sinds Versie 6 van DB2 UDB voor OS/390 is het mogelijk geworden om zogenaamde 'optimizer hints' te gebruiken. Via deze methode kan men het toegangspad, dat bepaald wordt door de DB2 optimizer, instellen. Normaalgesproken bepaalt de optimizer het toegangspad aan de hand van de statistische informatie die gevonden kan worden in de DB2 Catalog. Wanneer optimizer hints gebruikt worden, zal de optimizer een vooraf bepaald (en eventueel gemanipuleerd) toegangspad gebruiken. Het doel dat men wil bereiken met deze techniek, is vermijden dat 'slechte' toegangspaden gebruikt worden.

Figuur 1: Bepaling van het toegangspad zonder hints



Optimizer hints: Waarom?

Sinds het verschijnen van DB2 is de optimizer steeds krachtiger geworden. Het is meestal dan ook zo, dat het vastgestelde toegangspad inderdaad ook optimaal is. Soms beschikt de optimizer echter

niet over voldoende informatie om het meest optimale toegangspad te kiezen. In dit geval kan het ingrijpen van een applicatieprogrammeur/DBA noodzakelijk zijn.

De meerderheid van DB2 performance problemen (volgens sommigen is dit zelfs 65%) heeft te maken met slechte applicatie- en SQL design, op de voet gevolgd door inferieur database design. Indien men de performance van een slecht draaiende applicatie wil verhogen zal men in de eerste plaats aandacht moeten besteden aan de queries in de applicatie. Meer bepaald zal men de verschillende toegangspaden die de optimizer voor deze queries bepaald heeft moeten analyseren. Het is vanzelfsprekend dat in deze context eerst gedacht moet worden aan de meest voor de hand liggende redenen van slechte performance: verouderde statistieken (op te lossen door het RUNSTATS utility op regelmatige basis uit te voeren), geen , te weinig of verkeerde indexen (op te lossen door indexen te creëren of te droppen), slechte organisatie van de DB2 objecten (op te lossen door een REORG).

Indien deze echter niet aan de basis liggen van het probleem en de optimizer nog steeds het verkeerde toegangspad kiest, moet de programmeur of DBA zijn toevlucht nemen tot andere methodes.

Technieken om het toegangspad te beïnvloeden

Voor Versie 6 van DB2 waren er reeds allerlei methodes gedocumenteerd om de optimizer te 'misleiden'. Vele van deze technieken hebben als einddoel DB2 te sturen naar het gebruik (of juist het niet-gebruik) van een index. Een greep uit het aanbod:

- herschrijven van de query met behoud van functionaliteit: vele programmeurs of DBA's blijken erg creatief te zijn in het uitdenken van 'trucjes' om de optimizer in een andere richting te leiden. Ook het toevoegen van de OPTIMIZE FOR n ROWS clausule wordt veelvuldig toegepast. Hierbij dient wel opgemerkt te worden, dat in nieuwere releases van DB2 niet altijd hetzelfde resultaat wordt bekomen!

- het manipuleren van de DB2 Catalog: door een SQL UPDATE van kolommen van de Catalog tabellen waarin statistische informatie bewaard wordt, kan men bekomen dat de optimizer voor een welbepaald toegangspad zal kiezen. Het is waarschijnlijk overbodig op te merken dat het manipuleren van de Catalog een taak is die weggelegd is voor iemand met SYSADM bevoegdheid. Bovendien worden deze manipulaties ongedaan gemaakt door de uitvoering van een RUNSTATS.

Optimizer hints: Pro's en contra's

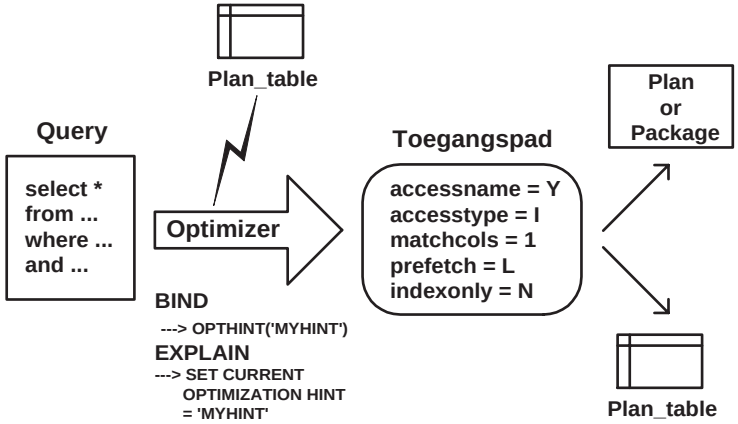
Bovenvermelde technieken bieden de applicatieprogrammeur of DBA vandaag de dag nog altijd veel mogelijkheden. Optimizer hints kunnen een alternatief betekenen voor deze 'misleidingstechnieken'. Maar optimizer hints zijn echter meer dan dat. Vele DBA's worden regelmatig geconfronteerd met een query die in een vorige release van DB2 geen performance probleem had, en bij een recentere release wel. Ook een (kleine) wijziging in het DB2 systeem, of het uitvoeren

van een RUNSTATS en daaropvolgende BIND/REBIND kan grote (en niet altijd positieve!) gevolgen hebben voor de performance van een query.

Optimizer hints geven de mogelijkheid om op elk moment terug te grijpen naar een vroeger (en op dat moment optimaler) toegangspad. Het is zelfs mogelijk om verscheidene toegangspaden te bewaren en de keuze van het toegangspad te laten afhangen van het moment. Op die manier zal het toegangspad altijd optimaal zijn.

Het werken met optimizer hints is echter niet zonder gevaren. Om optimaal gebruik te maken van alle voordelen van optimizer hints is een zeer grondige kennis van de werking van de optimizer vereist. Elk resultaat van een query waarbij optimizer hints gebruikt worden, moet grondig geanalyseerd worden. Ondoordacht gebruik van optimizer hints kan immers leiden tot performance verlies in de plaats van winst !

Figuur 2: Bepaling van het toegangspad met hints



Optimizer hints: Waar en wanneer?

Optimizer hints kunnen zowel gebruikt worden voor dynamische als statische embedded SQL statements. Voor statische SQL moet tijdens de BIND aangegeven worden welke optimizer hint gebruikt moet worden. Voor dynamische SQL wordt dit aangegeven net voor de query wordt uitgevoerd. Indien de hint effectief gebruikt wordt laat DB2 dit weten door een SQLCode + 394 terug te geven. In geval de hint werd overwogen, maar onbruikbaar bevonden werd door DB2, resulteert dit in een +395 SQLCode.

Om het gebruik van hints toe te laten zijn sinds Versie 6 van DB2 for OS/390 de volgende wijzigingen aangebracht:

- een nieuwe configuratieparameter laat toe om aan te duiden dat optimizer hints gebruikt kunnen worden (OPTMIZATION HINTS YES). De default waarde van deze parameter bij installatie is NO. In dit geval worden alle hints genegeerd.

- een nieuwe BIND parameter (OPTHINT = '*hint-id*') laat toe om tijdens de BIND/REBIND een unieke hint-naam mee te geven (statische embedded SQL)

- een nieuw special register statement (SET CURRENT OPTIMIZATION HINT = '*hint-id*') laat toe om voor de uitvoering van een SQL statement een unieke hint-naam vast te leggen (dynamische embedded SQL).

- toevoeging van twee nieuwe kolommen in de PLAN_TABLE: de OPTHINT kolom en de HINT_USED kolom.

Voor de volledige plantabel zie Plantabel op p. 13

OPTHINT (Char 8): laat toe om een toegangspad uniek te identificeren. Het toegangspad met een bepaalde OPTHINT-waarde zal gebruikt worden wanneer deze hint kenbaar wordt gemaakt tijdens de BIND of tijdens het runnen van de query. Deze waarde wordt aan de kolom toegekend door middel van een SQL UPDATE statement. In het vervolg van de tekst zullen we naar deze rij refereren als de HINT-rij.

HINT_USED (Char 8): indien een hint gebruikt werd, zal DB2 een rij toevoegen aan de PLAN_TABLE met de unieke naam van de OPTHINT in deze kolom. Deze rij wordt toegevoegd in het geval van statische SQL op het moment van een BIND/REBIND met EXPLAIN(YES) of voor dynamische SQL op moment dat een optimizer hint gebruikt wordt. Deze rij is een kopie van de HINT-rij uit de PLAN_TABLE en dient uitsluitend ter documentatie.

Bij veelvuldig gebruik van optimizer hints kan best een duplicate index gelegd worden op de PLAN_TABLE (op de kolommen : QUERYNO, APPLNAME, PROGNAME, Versie, COLLID en OPTHINT). Aangezien DB2 op deze kolommen zal zoeken telkens een optimizer hint gebruikt wordt, zal de creatie van een index een positief effect hebben op de performance van deze zoekacties.

Optimizer hints in de praktijk

Er bestaan twee scenario's om HINT-rijen in de PLAN_TABLE te verwerken:

- een 'self-made' toegangspad toevoegen aan de PLAN_TABLE door middel van een SQL INSERT actie.
- een bestaand toegangspad wijzigen in de PLAN_TABLE door middel van een SQL UPDATE actie.

Het tweede scenario verdient duidelijk voorkeur, gezien de complexiteit van de PLAN_TABLE. Wanneer een hint wordt meegegeven zal DB2 het toegangspad dat hiermee overeenkomt eerst valideren. Indien er ernstige problemen gedetecteerd worden (onbestaande index, inconsistente joinmethode,...), zal DB2 beslissen de hint niet te gebruiken. Dit zal resulteren in een SQLCode +395. Het toegangspad wordt dan op de gewone manier bepaald.

Voorbeeld

Om de toepassing van optimizer hints in de praktijk toe te lichten hebben wij gekozen voor volgend voorbeeld: productinformatie wordt opgeslagen in een tabel PRODUCTS. Elk product wordt ondermeer gekenmerkt door een productklasse(PRCLASS), een productnummer (PRNO), een productnaam (PRNAME) en een productiedatum (PRDATE). De tabel PRODUCTS heeft 3 indexen:

- een unieke clustered index op de primary key PRNO, PRCLASS(IND_PK)
- een unieke index op de PRNAME(IND_PRNAME)
- een niet-unieke index op de PRDATE(IND_PRDATE)

Beschouw nu de volgende query:

```
SELECT *
FROM PRODUCTS
WHERE PRCLASS < :PRCLASS
AND PRDATE BETWEEN :PRDATE1
AND :PRDATE2
```

Stel dat de optimizer in dit specifieke geval de voorkeur geeft aan het gebruik van de index IND_PK. Indien de hostvariabelen PRDATE1 en PRDATE2 echter in de praktijk meestal erg dicht bij elkaar liggen, zou het misschien beter zijn voor de performance dat de index IND_PRDATE gebruikt wordt. Hoe men dit kan bekomen door middel van optimizer hints wordt in de volgende paragrafen verder uitgewerkt

Scenario voor statische/dynamische SQL

- voeg in de bestaande applicatie MYAPPL aan de bewuste query een QUERYNO clausule toe:

```
SELECT *
FROM PRODUCTS
WHERE PRCLASS  <  :PRCLASS
AND PRNO < ?
AND PRDATE BETWEEN :PRDATE1
AND :PRDATE2
QUERYNO 1
```

- doe de BIND/REBIND van deze applicatie met EXPLAIN(YES), of voer een EXPLAIN uit in het geval van dynamische SQL. Voor de inhoud van de plantabel zie figuur 3 (1);

- wijzig eventueel het toegangspad door middel van een SQL UPDATE en markeer dit toegangspad als hint (MYHINT). Voor de inhoud van de plantabel na het uitvoeren van het UPDATE statement zie figuur 3 (2).

```
UPDATE PLAN_TABLE
SET ACCESSNAME =
'IND_PRDATE',
PREFETCH = 'L',
OPTHINT = 'MYHINT'
WHERE QUERYNO = 1
AND PROGNAME = 'MYAPPL'
```

- specificeer dat DB2 deze hint moet gebruiken:

```
BIND PLAN(MYPLAN) -
MEMBER(MYAPPL) -
LIBRARY('MYLIB') -
ACTION(REPLACE) -
VALIDATE(BIND) -
ISOLATION(CS) -
ACQUIRE(USE) -
RELEASE(COMMIT) -
OPTHINT('MYHINT') -
EXPLAIN(YES)
```

Voor statische SQL: geef dit mee tijdens de BIND/REBIND

Volledige syntax - zie Hints in statische sql op p. 18

```
ABISDB2 BIND SQL WARNING
USING ABISUSER AUTHORITY
PLAN=MYPLAN
DBRM=MYAPPL
STATEMENT=65
SQLCODE=394
SQLSTATE=01629
TOKENS=
```

De BIND/REBIND job zal in dit geval eindigen met een RC 4 en de volgende boodschap:

```
SET CURRENT OPTIMIZATION
HINT = 'MYHINT'
```

Voor dynamische SQL: maak gebruik van het special register.

Volledige syntax - zie Hints in dynamische sql op p. 14

Zie voor inhoud van de PLAN_TABLE voor beide gevallen figuur 3 (3).

Figuur 3: Inhoud van de plantabel bij het uitvoeren van de query zonder hints (1), na het wijzigen van het toegangspad in de plantabel (2), wanneer deze hint ook effectief gebruikt wordt door DB2, zowel voor statische als dynamische SQL (3).

QueryNo	TableName	AccessType	MatchCols	IndexOnly	AccessName	Prefetch	OptHints	HintUsed	
1	PRODUCTS	I	1	N	IND_PK	S			(1)
1	PRODUCTS	I	1	N	IND_PRDATE	L	MYHINT		(2)
1	PRODUCTS	I	1	N	IND_PRDATE	L	MYHINT		(3)
1	PRODUCTS	I	1	N	IND_PRDATE	L		MYHINT	

Besluit

Optimizer hints bieden de applicatieprogrammeur/DBA een bijkomende en zeer interessante mogelijkheid om het toegangspad dat de optimizer bepaalt, te beïnvloeden. Vermoedelijk zullen optimizer hints het meest geschikt zijn om terug te grijpen naar een vroeger toegangspad dat om de één of de andere reden performanter was.

Wel dient gezegd te worden dat het gebruik van optimizer hints zeer doordacht moet gebeuren. Eén van de vereisten is een zeer gefundeerde kennis van het hele DB2 systeem. Niet enkel kennis van de optimizer, maar ook kennis van de fysieke en logische design van de database is noodzakelijk. Deze taak is dan ook slechts weggelegd voor enkelen.

DOSSIER 7

Web services: een overzicht

Op 24 april 2001 lanceerde IBM DB2 UDB7.2. Met deze nieuwe versie van DB2 wordt het mogelijk om via Web service applicaties DB2 tabellen te benaderen. Via de DB2 XML Extender wordt het eveneens mogelijk om DB2 data als een XML document te ondervragen.

De DB2 XML Extender maakt het mogelijk om XML data te bewaren en op te vragen. Gegevens uit een XML document kunnen bewaard worden op twee manieren. Een eerste manier is de Column methode. Bij deze werkwijze wordt heel het XML document - inclusief tags - in zijn geheel bewaard in een zogenaamde XML kolom. Een tweede manier is de Collections methode waarbij de inhoud van het XML document geanalyseerd wordt. De gegevens worden zonder tags in de kolommen van (verschillende) tabellen opgeslagen. Opvragen van de gegevens in XML formaat is eveneens mogelijk.

De mapping tussen een XML document en de DB2 tabellen wordt gespecificeerd in een zogenaamde DAT file. Deze file kan zowel SQL statements bevatten om de DB2 tabellen te ondervragen als XPATH instructies. XPATH is een querytaal die gebruikt wordt om doorheen XML documenten te navigeren. De DB2 XML Extender beschikt over een aantal functies die het toelaten om via XPATH XML documenten - opgeslagen in XML kolommen - te ondervragen.

XML documenten die nog niet in DB2 zitten, kunnen geparsed en geanalyseerd worden via stored procedures. Parsen is het inlezen van een XML file. Deze stored procedures kunnen ook aangesproken worden via Web services. Het is ook mogelijk om zelf stored procedures te schrijven. Deze gaan dan met behulp van SQL queries de gegevens uit de DB2 tabellen halen. Het resultaat van zo'n stored procedure kan eveneens een XML (of een XHTML) document zijn. Vanzelfsprekend kunnen ook deze zelf gedefinieerde stored procedures ter beschikking gesteld worden van de internet gemeenschap. Dit gebeurt dan via zogenaamde UDDI sites waar men de input en de output beschrijft in een WSDL bestand. Voor diegene die deze procedures liever niet rechtstreeks aangesproken zien bestaat er ook nog de mogelijkheid om ze binnen de IBM WebSphere omgeving te gebruiken.

Bron: www.ibm.com/db2

Tom Avermaete (ABIS)

1,2,3,4,5....en wat daarna? Over identity kolommen (i)

Katrien Platteborze (ABIS)

Stel u een doorsnee alledaagse actie voor die uitgevoerd wordt in een relationele database omgeving: een nieuwe rij toevoegen aan een tabel. Inderdaad alledaags - een insert statement - eenvoudiger kan haast niet. Maar toch, misschien is dit eenvoudig insert statement wel de oorzaak van het 'wij-weten-hoe-het-moet' gevoel. U kent het probleem: welke waarde voorzien we voor de primary key? Als we een volgende rij willen toevoegen, met een nieuwe waarde voor de primary key, dan start de zoektocht naar de vorige waarde.

Tot voor kort werden hiervoor applicatie specifieke oplossingen gezocht. Maar vanaf DB2 UDB voor OS/390 V7 kan het anders, met behulp van een identity kolom. Een identity kolom is een kolom waarvoor DB2 automatisch een numerieke sequentiële waarde voorziet wanneer een rij toegevoegd wordt aan de tabel. DB2 doet dit op basis van informatie die hij in de catalog bijhoudt. Identity kolommen kunnen dus gebruikt worden om unieke primary key waarden te genereren. Dit vermijdt een aantal concurrency en performance problemen die kunnen voorkomen wanneer de sequentiële waarden door de applicatie bepaald worden. Uniekheid van de gegenereerde waarden kan verzekerd worden door een unieke index aan te maken op de identity kolom.

Aanmaken van een identity kolom

Willen we ook zelf waarden toevoegen, of is het alleen DB2 die waarden voorziet? Moeten de waarden echt uniek zijn of mogen ze herbruikt worden? Wat is de start waarde en met hoeveel wordt deze verhoogd om een nieuwe waarde te bepalen? Wensen we een maximum of een minimum in te stellen? Misschien willen we wel aflopende waarden i.p.v. oplopende. De waarden die DB2 voorziet en wat we zelf met de identity kolom kunnen doen is afhankelijk van de definitie van de identity kolom op het create of alter table statement.

Bij het aanmaken van een identity kolom moeten we in eerste instantie met de volgende zaken rekening houden:

- Per tabel is er maar één identity kolom mogelijk.
- Het datatype van de kolom moet smallint, integer of decimal (met scale 0) zijn.
- De kolom wordt automatisch met not null gedefinieerd.
- De kolom mag niet met een default gedefinieerd worden.

Een kolom wordt een identity kolom door er GENERATED ALWAYS AS IDENTITY of GENERATED BY DEFAULT AS IDENTITY aan toe te voegen. GENERATED ALWAYS betekent dat DB2 altijd een waarde genereert en dat je geen waarde mag inserten. Doe je dit toch dan krijg je een fout melding. GENERATED BY DEFAULT betekent dat DB2 enkel een waarde voorziet wanneer jij het niet doet. Daarnaast kunnen er nog een aantal opties toegevoegd worden.

```
CREATE TABLE tableA
(PKeyColumn  SMALLINT
PRIMARY KEY
GENERATED ALWAYS AS
IDENTITY
,...
```

In dit eerste voorbeeld worden er geen specifieke opties gebruikt waardoor DB2 overal defaults gaat gebruiken

```
CREATE TABLE tableA
(PKeyColumn  smallint  NOT
NULL GENERATED ALWAYS AS
IDENTITY
(START WITH 1,
INCREMENT BY 1, CACHE 20,
NO CYCLE, MAXVALUE 32768,
MINVALUE -32767)
, ....
```

Het uitvoeren van dit tweede create statement, waarbij alle opties ingevuld werden met de default waarden die DB2 kiest, levert exact dezelfde tabel op.

De meeste opties zoals START WITH 1, INCREMENT BY 1, MINVALUE -32768, MAXVALUE 32767 zijn self explaining. CYCLE of NO CYCLE specificeert of DB2 nog waarden mag generen wanneer de maximum of minimum (wanneer increment een negatief getal is) waarde bereikt werd. DB2 herbegint dan bij de respectievelijk minimum of maximum waarde. Dit betekent dat dubbele waarden kunnen voorkomen in de identity kolom, tenzij er een unieke index gedefinieerd werd. Met CACHE kan men aanduiden of men waarden (en hoeveel waarden) wilt prealloceren in het geheugen. Dit om de performance te verbeteren. De verschillende mogelijkheden zijn: CACHE 20, CACHE geheel getal, NO CACHE. Het gebruik van de CACHE optie kan in bepaalde omstandigheden anomalieën veroorzaken zoals het wegvallen van een aantal waarden of het niet op één volgend toekennen van waarden.

Het resultaat kan men terugvinden in SYSIBM.SYSSEQUENCESDEP en SYSIBM.SYSSEQUENCES zie figuur 1 en 2.

Figuur 1: Inhoud van SYSIBM.SYSSEQUENCESDEP

bsequenceid	dname	dcolname
258	tableA	Pkeycolumn
259	tableB	Column

Figuur 2: Inhoud van SYSIBM.SYSSEQUENCES

sequenceid	maxvalue	minvalue	max assigned val	increment	start	cycle	cache
258	32767	-32768	------	1	1	N	20
259	99999	1000	------	5	1000		N

```
CREATE TABLE tableB
(Column  INTEGER NOT NULL
GENERATED BY DEFAULT AS
IDENTITY
(START WITH 1000,
INCREMENT BY 5,
NO CACHE,
CYCLE,
MAXVALUE 99999,
MINVALUE 1000), ...
```

In het volgende voorbeeld wordt GENERATED BY DEFAULT en een aantal opties gebruikt. Het resultaat vindt men eveneens terug in figuur 1 en 2.

Opmerking: Indien je één van de opties van een bestaande identity kolom wenst te wijzigen dan moet de tabel gedropt en terug aangemaakt worden

Toevoegen van data in een identity kolom

Zoals we reeds vermeld hadden is het al dan niet kunnen toevoegen van waarden afhankelijk van de definitie van de identity kolom: met GENERATED ALWAYS of met GENERATED BY DEFAULT.

Wanneer we aan tabelA 21 rijen toevoegen wordt de maxassignedval in sysibm.syssequences op 40 gezet. Met CACHE 20 worden bij het toevoegen van de eerste rij 20 waarden gealloceerd (maxassignedval staat op 20). Wanneer deze opgebruikt zijn worden de volgende 20 waarden gealloceerd. Dit is in dit voorbeeld het geval wanneer we de 21ste rij inserten.

Wanneer we aan tabelB 21 rijen toevoegen zal de maxassignedval 1100 bedragen.

Als om één of andere reden de waarden niet aaneensluitend zijn is het onmogelijk dit te herstellen, tenzij jezelf kan inserten (GENERATED BY DEFAULT).Ik voeg bijvoorbeeld 10 rijen toe aan tabel A. Ik verwijder 9 rijen. Ik voeg terug een rij toe. De genereerde waarde zal 11 zijn ook al had ik misschien graag gehad dat het 2 was geweest. Ook een update van de catalog tabel is in dit kader niet mogelijk.

In een volgende editie wordt verder ingegaan op het gebruik van de identity kolom bij load operaties en het gebruik van de cache optie.

CURSUSPLANNING SEP-DEC 2002

DB2 for OS 390, een totaaloverzicht	1625 EUR	23-27/09 (L), 14-18/10 (W), 04-08/11 (L), 18-22/11 (W), 09-13/12 (L)
DB2 UDB, een totaaloverzicht	1625 EUR	18-19 & 27-29/11 (W)
RDBMS concepten	325 EUR	23/09 (L),14/10 (W), 04/11 (L), 18/11 (W), 09/12 (L)
Basiskennis SQL	325 EUR	24/09 (L), 15/10 (W), 05/11 (L) 19/11 (W), 10/12 (L)
DB2 for OS/390 basiscursus	975 EUR	25-27/09 (L), 16-18/10 (W), 06-08/11 (L), 20-22/11 (W), 11-13/12 (L)
DB2 UDB basiscursus	975 EUR	25-27/09 (L), 27-29/11 (W)
DB2 UDB concepten	375 EUR	12/11 (W), 03/12 (L)
SQL workshop	700 EUR	07-08/10 (L), 24-25/10 (W), 16-17/12 (L)
DB2 applicatieprogrammering voor gevorderden	1050 EUR	21-23/10 (W), 04-06/12 (L)
DB2 for OS/390: SQL performance	1200 EUR	13-15/11 (W), 16-18/12 (L)
Fysiek ontwerp van relationele databases	700 EUR	04-05/11 (L)
DB2 database administratie	1600 EUR	23-26/09 (W), 18-21/11 (L)
DB2 UDB database administratie	1600 EUR	12-15/11 (L)
DB2 UDB systeembeheer en performance	400 EUR	27/09 (W), 25/11 (L)
DB2 UDB for OS/390 V7 upgrade voor ontwikkelaars	375 EUR	25/10 (L), 05/12 (W)
DB2 UDB en zijn extenders: XML en text search	200 EUR	04/11 (W), 16/12 (L)
DB2 UDB integratie met MQSeries	200 EUR	04/11 (W), 16/12 (L)

Plaats: L = Leuven; W = Woerden
Details, andere data en bijkomende cursussen: www.abis.be

Postbus 220
Diestsevest 32
BE-3000 Leuven
Tel. 016/245610
Fax 016/245691
training@abis.be

abis
TRAINING & CONSULTING

Postbus 122
Pelmolenlaan 1-K
NL-3440 AC Woerden
Tel. 0348-435570
Fax 0348-432493
training@abis.be

Bijlagen

Plantabel

```
QUERYNO                         INTEGER NOT NULL,
QBLOCKNO                        SMALLINT NOT NULL,
APPLNAME                        CHAR(8) NOT NULL,
PROGNAME                        CHAR(8) NOT NULL,
PLANNO                          SMALLINT NOT NULL,
METHOD                          SMALLINT NOT NULL,
CREATOR                         CHAR(8) NOT NULL,
TNAME                           CHAR(18) NOT NULL,
TABNO                           SMALLINT NOT NULL,
ACCESSTYPE                      CHAR(2) NOT NULL,
MATCHCOLS                       SMALLINT NOT NULL,
ACCESSCREATOR                   CHAR(8) NOT NULL,
ACCESSNAME                      CHAR(18) NOT NULL,
INDEXONLY                       CHAR(1) NOT NULL,
SORTN_UNIQ                      CHAR(1) NOT NULL,
SORTN_JOIN                      CHAR(1) NOT NULL,
SORTN_ORDERBY                   CHAR(1) NOT NULL,
SORTN_GROUPBY                   CHAR(1) NOT NULL,
SORTC_UNIQ                      CHAR(1) NOT NULL,
SORTC_JOIN                      CHAR(1) NOT NULL,
SORTC_ORDERBY                   CHAR(1) NOT NULL,
SORTC_GROUPBY                   CHAR(1) NOT NULL,
TSLOCKMODE                      CHAR(3) NOT NULL,
TIMESTAMP                       CHAR(16) NOT NULL,
REMARKS                         VARCHAR(254) NOT NULL,
--25-column-format--
PREFETCH                        CHAR(1) NOT NULL,
COLUMN_FN_EVAL                  CHAR(1) NOT NULL,
MIXOPSEQ                        SMALLINT NOT NULL,
--28-column-format--
VERSION                         VARCHAR(64) NOT NULL,
COLLID                          CHAR(18) NOT NULL,
--30-column-format--
ACCESS_DEGREE                   SMALLINT,
ACCESS_PGROUP_ID                SMALLINT,
JOIN_DEGREE                     SMALLINT,
JOIN_PGROUP_ID                  SMALLINT,
--35-column-format--
SORTC_PGROUP_ID                 SMALLINT,
SORTN_PGROUP_ID                 SMALLINT,
PARALLELISM_MODE                CHAR(1),
MERGE_JOIN_COLS                 SMALLINT,
CORRELATION_NAME                CHAR(18),
PAGE_RANGE                      CHAR(1) NOT NULL,
JOIN_TYPE                       CHAR(1) NOT NULL,
GROUP_MEMBER                    CHAR(8) NOT NULL,
IBM_SERVICE_DATA                VARCHAR(254) NOT NULL,
--43-column-format--
WHEN_OPTIMIZE                   CHAR(1)   NOT NULL WITH DEFAULT,
QBLOCK_TYPE                     CHAR(6)   NOT NULL WITH DEFAULT,
BIND_TIME                       TIMESTAMP NOT NULL WITH DEFAULT,
--46-column-format--
OPTHINT                         CHAR(8) NOT NULL WITH DEFAULT,
HINT_USED                       CHAR(8) NOT NULL WITH DEFAULT,
PRIMARY_ACCESSTYPE              CHAR(1) NOT NULL WITH DEFAULT)
--49-column-format--
```

Hints in dynamische sql

```
*****************************************
1. CREATIE VAN DE TABLE PRODUCTS EN INDEXEN
*****************************************

CREATE TABLESPACE TBSPROD
IN ABISDB
  ;
CREATE TABLE PRODUCTS
(
PRCLASS  CHAR(4) NOT NULL,
PRNO    CHAR(2)  NOT NULL,
PRNAME   CHAR(50)   NOT NULL,
PRDATE   DATE   NOT NULL,
PRSTATUS CHAR(2),
PRDESC    CHAR(100),
PRIMARY KEY (PRCLASS,PRNO)
)
IN ABISDB.TBSPROD
;

CREATE UNIQUE INDEX IND_PK ON PRODUCTS (PRCLASS,PRNO) CLUSTER;
CREATE UNIQUE INDEX IND_PRNAME ON PRODUCTS(PRNAME);
CREATE        INDEX IND_PRDATE ON PRODUCTS(PRDATE);



*********************************
2. OPVULLEN VAN DE TABLE PRODUCTS
*********************************

INSERT INTO PRODUCTS  (PRNO,PRCLASS,PRNAME,PRSTATUS,PRDATE,PRDESC)
SELECT  PRNO, PRCLASS, PRNAME, PRSTATUS, PRSTDATE,PRTEMP
FROM TTTPRODUCTS
;
SELECT *
FROM PRODUCTS



*******************
3. CATALOG TABELLEN
*******************

--SYSIBM.SYSTABLESPACE
SELECT  NAME, NTABLES, INTEGER(NACTIVEF) AS NACTIVEF
FROM  SYSIBM.SYSTABLESPACE
WHERE (DBNAME = 'TBDB031')
AND NAME LIKE 'TBSPROD%'
ORDER BY NAME
-----------------------------------------------------
NAME      NTABLES     NACTIVEF
TBSPROD         1        50000
-----------------------------------------------------
--SYSIBM.SYSTABLES
SELECT  TSNAME, SUBSTR(NAME, 1, 14) AS TABLE,
        INTEGER(CARDF) AS CARDF, NPAGES,
        PCTPAGES AS PCT,
        RECLENGTH
FROM  SYSIBM.SYSTABLES
WHERE (DBNAME = 'TBDB031')
AND TSNAME LIKE 'TBSPROD%'
```

```
AND NAME LIKE 'PROD%'
ORDER BY TSNAME, TABLE
-------------------------------------------------------
TSNAME    TABLE         CARDF       NPAGES    PCT  RECLENGTH
TBSPROD   PRODUCTS      500000      50000     100        172
-------------------------------------------------------
-- SYSIBM.SYSCOLUMNS
SELECT  SUBSTR(C.TBNAME, 1, 13) AS TABLE,
        SUBSTR(C.NAME, 1, 10) AS COLUMN
        , INTEGER(COLCARDF) AS CARDF, HIGH2KEY, LOW2KEY
FROM  SYSIBM.SYSCOLUMNS C, SYSIBM.SYSTABLES T
WHERE (C.TBNAME = T.NAME)
AND (C.TBCREATOR = T.CREATOR)
AND (DBNAME = 'TBDB031')
AND T.NAME LIKE 'PROD%'
ORDER BY 1, COLNO
-------------------------------------------------------
TABLE          COLUMN            CARDF  HIGH2KEY  LOW2KEY
PRODUCTS       PRCLASS           10000  9999      0000
PRODUCTS       PRNO                100  99        01
PRODUCTS       PRNAME           500000  vegetabl  agricult
PRODUCTS       PRDATE             1500  --        --
PRODUCTS       PRSTATUS             -1
PRODUCTS       PRDESC               -1
-------------------------------------------------------
-- SYSIBM.SYSINDEXES
SELECT  SUBSTR(NAME, 1, 8) AS INDEX,
        SUBSTR(COLNAME, 1, 10) AS COLUMN,
        INTEGER(FIRSTKEYCARDF) AS FIRSTKEYC,
        INTEGER(FULLKEYCARDF) AS FULLKEYC,
        INTEGER(CLUSTERRATIOF * 100) AS CLRATIOF,
        NLEAF, NLEVELS AS NLVLS
FROM  SYSIBM.SYSINDEXES, SYSIBM.SYSKEYS
WHERE (IXNAME = NAME)
AND (IXCREATOR = CREATOR)
AND (DBNAME = 'TBDB031')
AND (IXNAME LIKE 'IND_PR%' OR IXNAME LIKE 'IND_PK%')
ORDER BY INDEX, 2, COLSEQ
-------------------------------------------------------
INDEX    COLUMN  FIRSTKEYC FULLKEYC CLRATIOF  NLEAF NLVLS
IND_PK   PRCLASS     10000   500000      100   1666     3
IND_PK   PRNO        10000   500000      100   1666     3
IND_PRDA PRDATE       1500     1500       10   1875     3
IND_PRNA PRNAME     500000   500000       10   7000     4
-------------------------------------------------------



*****************************
4. OPVULLEN VAN DE PLAN_TABLE
*****************************

DELETE FROM PLAN_TABLE;
DELETE FROM DSN_STATEMNT_TABLE;
EXPLAIN PLAN
SET QUERYNO = 1
FOR
SELECT *
FROM PRODUCTS
WHERE PRCLASS < ? AND PRNO < ?
AND PRDATE BETWEEN ? AND ?
--HIER KIEST DE OPTIMIZER VOOR EEN ACCESSPATH VIA INDEX --OP IND_PK
```

```
--MAAR ALS DE TWEE DATUMS ERG DICHT TEGEN ELKAAR LIGGEN --IS DE INDEX
OP DE PRDATE BETER ---> ZIE 2
;

EXPLAIN PLAN
SET QUERYNO = 2
FOR
SELECT *
FROM PRODUCTS
WHERE PRCLASS < '1000' AND PRNO < '2'
AND PRDATE BETWEEN '01.01.1999' AND '02.01.1999'
;
SELECT  SUBSTR(DIGITS(QUERYNO), 8, 3) AS Q#,
        SUBSTR(DIGITS(QBLOCKNO), 5, 1) AS B#,
        SUBSTR(DIGITS(PLANNO), 5, 1) AS P#,
        SUBSTR(DIGITS(METHOD), 5, 1) AS MTH,
        SUBSTR(TNAME, 1, 8) AS TBNAME,
        SUBSTR(DIGITS(TABNO), 5, 1) AS T#,
        SUBSTR(ACCESSTYPE, 1, 2) AS A,
        SUBSTR(DIGITS(MATCHCOLS), 5, 1) AS MC,
        SUBSTR(ACCESSNAME, 1, 8) AS IXNAME,
        SUBSTR(INDEXONLY, 1, 1) AS IO,
       SORTN_UNIQ!!SORTN_JOIN!!SORTN_ORDERBY!!SORTN_GROUPBY AS N_UJOG,
       SORTC_UNIQ!!SORTC_JOIN!!SORTC_ORDERBY!!SORTC_GROUPBY AS C_UJOG,
        SUBSTR(PREFETCH, 1, 1) AS PF,
        OPTHINT,
        HINT_USED,
        TSLOCKMODE,
        SUBSTR(TIMESTAMP, 7, 8)
FROM  PLAN_TABLE
ORDER BY 1, 2, 3, 17, 14
;
------------------------------------------------------------------------
Q#   B#  P#  MTH  TBNAME    T#  A   MC  IXNAME    IO  N_UJOG C_UJOG PF
001  1   1   0    PRODUCTS  1   I   1   IND_PK    N     NNNN   NNNN  S
002  1   1   0    PRODUCTS  1   I   1   IND_PRDA  N     NNNN   NNNN  L
------------------------------------------------------------------------
-- COST ESTIMATION
SELECT  SUBSTR(DIGITS(QUERYNO), 7, 4) AS Q9,
        STMT_TYPE AS S_TYPE,
        COST_CATEGORY AS CC,
        SUBSTR(DIGITS(PROCSU), 1, 10) AS SU,
        REASON
FROM  DSN_STATEMNT_TABLE
ORDER BY 1, EXPLAIN_TIME
;
------------------------------------------------------------------------
Q9    S_TYPE  CC  SU          REASON
0001  SELECT  B   0000000054  HOST VARIABLES
0002  SELECT  A   0000000106
------------------------------------------------------------------------



**************************
5. UPDATE VAN DE PLAN_TABLE
**************************

UPDATE PLAN_TABLE
SET OPTHINT = 'MYHINT'
WHERE QUERYNO = 2
;
SELECT  SUBSTR(DIGITS(QUERYNO), 8, 3) AS Q#,
```

```
        SUBSTR(DIGITS(QBLOCKNO), 5, 1) AS B#,
        SUBSTR(DIGITS(PLANNO), 5, 1) AS P#,
        SUBSTR(DIGITS(METHOD), 5, 1) AS MTH,
        SUBSTR(TNAME, 1, 8) AS TBNAME,
        SUBSTR(DIGITS(TABNO), 5, 1) AS T#,
        SUBSTR(ACCESSTYPE, 1, 2) AS A,
        SUBSTR(DIGITS(MATCHCOLS), 5, 1) AS MC,
        SUBSTR(ACCESSNAME, 1, 8) AS IXNAME,
        SUBSTR(INDEXONLY, 1, 1) AS IO,
        SUBSTR(PREFETCH, 1, 1) AS PF,
        OPTHINT,
        HINT_USED,
        TSLOCKMODE,
        SUBSTR(TIMESTAMP, 7, 8)
FROM  PLAN_TABLE
ORDER BY 1, 2, 3
;
-----------------------------------------------------------------------
Q#   B#  P#  MTH  TBNAME    T#  A   MC  IXNAME    IO  PF  OPTHINT HINT
001  1   1   0    PRODUCTS  1   I   1   IND_PK    N   S
002  1   1   0    PRODUCTS  1   I   1   IND_PRDA  N   L   MYHINT
-----------------------------------------------------------------------



**********************
6.GEBRUIK VAN DE HINT
**********************

DELETE FROM DSN_STATEMNT_TABLE
;

SET CURRENT OPTIMIZATION HINT = 'MYHINT'
;
EXPLAIN PLAN
SET QUERYNO = 2
FOR
SELECT *
FROM PRODUCTS
WHERE PRCLASS < ? AND PRNO < ?
AND PRDATE BETWEEN ? AND ?
;
-----------------------------------------------------------------------
SQLCODE = 394, WARNING:  USER SPECIFIED OPTIMIZATION HINTS USED DURING
ACCESS PATH SELECTION
SQLSTATE   = 01629 SQLSTATE RETURN CODE
SQLERRP    = DSNXOPTH SQL PROCEDURE DETECTING ERROR
SQLERRD    = 20  0  0  1146113695  0  0 SQL DIAGNOSTIC INFORMATION
SQLERRD    = X'00000014'  X'00000000'  X'00000000'  X'44504E9F'
X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
-----------------------------------------------------------------------
SELECT  SUBSTR(DIGITS(QUERYNO), 8, 3) AS Q#,
        SUBSTR(DIGITS(QBLOCKNO), 5, 1) AS B#,
        SUBSTR(DIGITS(PLANNO), 5, 1) AS P#,
        SUBSTR(DIGITS(METHOD), 5, 1) AS MTH,
        SUBSTR(TNAME, 1, 8) AS TBNAME,
        SUBSTR(DIGITS(TABNO), 5, 1) AS T#,
        SUBSTR(ACCESSTYPE, 1, 2) AS A,
        SUBSTR(DIGITS(MATCHCOLS), 5, 1) AS MC,
        SUBSTR(ACCESSNAME, 1, 8) AS IXNAME,
        SUBSTR(INDEXONLY, 1, 1) AS IO,
        SUBSTR(PREFETCH, 1, 1) AS PF,
        OPTHINT,HINT_USED,
```

```
        TSLOCKMODE,
        SUBSTR(TIMESTAMP, 7, 8)
FROM  PLAN_TABLE
ORDER BY 1, 2, 3
;
-----------------------------------------------------------------------
Q#   B#  P#  MTH  TBNAME    T#  A   MC  IXNAME    IO  PF OPTHINT HUSED
001  1   1   0    PRODUCTS  1   I   1   IND_PK    N   S
002  1   1   0    PRODUCTS  1   I   1   IND_PRDA  N   L  MYHINT
002  1   1   0    PRODUCTS  1   I   1   IND_PRDA  N   L          MYHINT
-----------------------------------------------------------------------
-- COST ESTIMATION
SELECT  SUBSTR(DIGITS(QUERYNO), 7, 4) AS Q9,
        STMT_TYPE AS S_TYPE,
        COST_CATEGORY AS CC,
        SUBSTR(DIGITS(PROCSU), 1, 10) AS SU,
        REASON
FROM  DSN_STATEMNT_TABLE
ORDER BY 1, EXPLAIN_TIME
;
-----------------------------------------------------------------------
Q9    S_TYPE  CC  SU          REASON
0002  SELECT  B   0000001086  HOST VARIABLES
```

Hints in statische sql

```
============================
1. COBOL APPLICATIE OHAPPL
============================

       IDENTIFICATION DIVISION.
      ************************
       PROGRAM-ID.    OHAPPL.
      *
      ****************************************************************
      ***  ABIS N.V. LEUVEN, BELGIUM. **
      ****************************************************************
      *
       ENVIRONMENT DIVISION.
      **********************
       DATA DIVISION.
      *===============
       FILE SECTION.
      *-------------
      *
       WORKING-STORAGE SECTION.
      *------------------------
      *
      * SQL DECLARES.
      *--------------
           EXEC SQL
             INCLUDE SQLCA
           END-EXEC.


           EXEC SQL DECLARE PRODUCTS TABLE
           ( PRCLASS                       CHAR(4) NOT NULL,
             PRNO                          CHAR(2) NOT NULL,
             PRNAME                        CHAR(50) NOT NULL,
             PRDATE                        DATE NOT NULL,
             PRSTATUS                      CHAR(2),
             PRDESC                        CHAR(100)
           ) END-EXEC.
```

```
************************************************************************
      * COBOL DECLARATION FOR TABLE PRODUCTS                           *
************************************************************************
       01  DCLPRODUCTS.
           10 PRCLASS              PIC X(4).
           10 PRNO                 PIC X(2).
           10 PRNAME               PIC X(50).
           10 PRDATE               PIC X(10).
************************************************************************
     * THE NUMBER OF COLUMNS DESCRIBED BY THIS DECLARATION IS 6       *
************************************************************************

      * HOST VARIABLES.
      *----------------
       01  RET-CODE       PIC S999 SIGN LEADING SEPARATE.
       01  PRDATE1        PIC X(10).
       01  PRDATE2        PIC X(10).

      * CURSOR DECLARATIE
      ********************
           EXEC SQL
              DECLARE CURPRODUCTS CURSOR FOR
                 SELECT *
                 FROM PRODUCTS
                 WHERE PRCLASS < :PRCLASS
                 AND PRNO < : PRNO
                 AND PRDATE BETWEEN :PRDATE1 AND :PRDATE2
                 QUERYNO 1
           END-EXEC

       PROCEDURE DIVISION.
      ********************
       MAIN.
           MOVE '1000' TO PRCLASS
           MOVE '1' TO PRNO
           MOVE '01.01.1999' TO PRDATE1
           MOVE '02.01.1999' TO PRDATE2

           EXEC SQL
              OPEN CURPRODUCTS
           END-EXEC
           MOVE SQLCODE TO RET-CODE
           DISPLAY 'OPEN CURSOR IS DE SQLCODE : ' RET-CODE

           EXEC SQL
              FETCH CURPRODUCTS
              INTO :DCLPRODUCTS
           END-EXEC
           MOVE SQLCODE TO RET-CODE
           DISPLAY 'NA EERSTE FETCH IS DE SQLCODE : ' RET-CODE

           PERFORM UNTIL SQLCODE NOT = 0
               DISPLAY 'EERSTE PRODUCT' DCLPRODUCTS
               EXEC SQL
                   FETCH CURPRODUCTS
                   INTO : DCLPRODUCTS
               END-EXEC
           END-PERFORM
           MOVE SQLCODE TO RET-CODE
```

```
           DISPLAY 'NA EERSTE FETCH IS DE SQLCODE : ' RET-CODE

           GOBACK.
```

```
====================
2. JCL VOOR DE BIND
====================

//BIND    EXEC PGM=IKJEFT01,DYNAMNBR=20
//STEPLIB DD   DSN=SYS.ABISDB2.DSNLOAD,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSOUT  DD   SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(ABISDB2) RETRY(0) TEST(0)
BIND PLAN(MYPLAN)  -
     MEMBER(OHAPPL) -
     LIBRARY('ABIS.DB2.DBRM') -
     ACTION(REPLACE)  -
     VALIDATE(BIND) -
     ISOLATION(CS) -
     ACQUIRE(USE) -
     RELEASE(COMMIT) -
     EXPLAIN(YES)
END
/*
```

```
====================================
3. SELECT VOOR RAADPLEGING PLAN_TABLE
====================================

SELECT  SUBSTR(DIGITS(QUERYNO), 8, 3) AS Q#,
        SUBSTR(DIGITS(QBLOCKNO), 5, 1) AS B#,
        SUBSTR(DIGITS(PLANNO), 5, 1) AS P#,
        SUBSTR(DIGITS(METHOD), 5, 1) AS MTH,
        SUBSTR(TNAME, 1, 8) AS TBNAME,
        SUBSTR(DIGITS(TABNO), 5, 1) AS T#,
        SUBSTR(ACCESSTYPE, 1, 2) AS A,
        SUBSTR(DIGITS(MATCHCOLS), 5, 1) AS MC,
        SUBSTR(ACCESSNAME, 1, 8) AS IXNAME,
        SUBSTR(INDEXONLY, 1, 1) AS IO,
        SUBSTR(PREFETCH, 1, 1) AS PF,
        OPTHINT,
        HINT_USED,
        TSLOCKMODE,
        SUBSTR(TIMESTAMP, 7, 8)
FROM  PLAN_TABLE
ORDER BY 1, 2, 3
;
------------------------------------------------------------------------
Q#  B# P#  MTH  TBNAME    T#  A MC IXNAME    IO  PF OPTHINT HINTU
001 1  1   0    PRODUCTS  1   I 1  IND_PK     N   S
------------------------------------------------------------------------
select * from plan_table
order by 1;
-- COST ESTIMATION
SELECT  SUBSTR(DIGITS(QUERYNO), 7, 4) AS Q9,
        STMT_TYPE AS S_TYPE,
        COST_CATEGORY AS CC,
        SUBSTR(DIGITS(PROCSU), 1, 10) AS SU,
```

```
        REASON
FROM  DSN_STATEMNT_TABLE
ORDER BY 1, EXPLAIN_TIME
-------------------------------------------------------------------------
Q9    S_TYPE  CC  SU          REASON
0001  SELECT  B   00000054  HOST VARIABLES



============================
4. UPDATE VAN DE PLAN_TABLE
============================

UPDATE PLAN_TABLE
SET OPTHINT = 'MYHINT',
ACCESSNAME = 'IND_PRDATE',
PREFETCH = 'L'
WHERE QUERYNO = 1
AND PROGNAME = 'OHAPPL'
;
SELECT  SUBSTR(DIGITS(QUERYNO), 8, 3) AS Q#,
        SUBSTR(DIGITS(QBLOCKNO), 5, 1) AS B#,
        SUBSTR(DIGITS(PLANNO), 5, 1) AS P#,
        SUBSTR(DIGITS(METHOD), 5, 1) AS MTH,
        SUBSTR(TNAME, 1, 8) AS TBNAME,
        SUBSTR(DIGITS(TABNO), 5, 1) AS T#,
        SUBSTR(ACCESSTYPE, 1, 2) AS A,
        SUBSTR(DIGITS(MATCHCOLS), 5, 1) AS MC,
        SUBSTR(ACCESSNAME, 1, 8) AS IXNAME,
        SUBSTR(INDEXONLY, 1, 1) AS IO,
        SUBSTR(PREFETCH, 1, 1) AS PF,
        OPTHINT,
        HINT_USED,
        TSLOCKMODE,
        SUBSTR(TIMESTAMP, 7, 8)
FROM  PLAN_TABLE
ORDER BY 1, 2, 3
;
-------------------------------------------------------------------------
Q#   B#  P#  MTH  TBNAME    T#  A   MC  IXNAME    IO  PF  OPTHINT HINT
001  1   1   0    PRODUCTS  1   I   1   IND_PRDA  N   L   MYHINT
-------------------------------------------------------------------------



============================
5. BIND MET GEBRUIK VAN HINT
============================

//BIND    EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT,LKED)
//STEPLIB DD   DSN=SYS.ABISDB2.DSNLOAD,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSOUT  DD   SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(ABISDB2) RETRY(0) TEST(0)
BIND PLAN(MYPLAN)  -
     MEMBER(OHAPPL) -
     LIBRARY('ABIS.DB2.DBRM') -
     ACTION(REPLACE)  -
     VALIDATE(BIND) -
     ISOLATION(CS) -
     ACQUIRE(USE) -
     OPTHINT('MYHINT') -
```

```
     RELEASE(COMMIT) -
     EXPLAIN(YES)
END
/*
```

```
======================
6. OUTPUT VAN DE BIND
======================
```

```
 READY
 DSN SYSTEM(ABISDB2) RETRY(0) TEST(0)
 DSN
 BIND PLAN(MYPLAN)
 MEMBER(OHAPPL)
 LIBRARY('ABIS.DB2.DBRM')
 ACTION(REPLACE)
 VALIDATE(BIND)
 ISOLATION(
 CS)
 ACQUIRE(USE)
 OPTHINT('MYHINT')
 RELEASE(COMMIT)
 EXPLAIN(YES)
 DSNX105I =ABISDB2 BIND SQL WARNING
            USING ABISUSER AUTHORITY
            PLAN=MYPLAN
            DBRM=OHAPPL
            STATEMENT=61
            SQLCODE=394
            SQLSTATE=01629
            TOKENS=··
 DSNT252I =ABISDB2 DSNTBCM1 BIND OPTIONS FOR PLAN MYPLAN
            ACTION        REPLACE
            OWNER         ABISUSER
            VALIDATE      BIND
            ISOLATION     CS
            ACQUIRE       USE
            RELEASE       COMMIT
            EXPLAIN       YES
            DYNAMICRULES  RUN
 DSNT253I =ABISDB2 DSNTBCM1 BIND OPTIONS FOR PLAN MYPLAN
            NODEFER       PREPARE
            CACHESIZE     1024
            QUALIFIER     ABISUSER
            CURRENTSERVER
            CURRENTDATA   YES
            DEGREE        1
            SQLRULES      DB2
            DISCONNECT    EXPLICIT
            NOREOPT       VARS
            KEEPDYNAMIC   NO
            IMMEDWRITE    NO
            DBPROTOCOL    PRIVATE
            OPTHINT       MYHINT
            PATH
 DSNT200I =ABISDB2  BIND FOR PLAN MYPLAN SUCCESSFUL
 DSN
 END
 READY
 END
```

```
==========================
7. SELECT VAN DE PLAN_TABEL
==========================

SELECT  SUBSTR(DIGITS(QUERYNO), 8, 3) AS Q#,
        SUBSTR(DIGITS(QBLOCKNO), 5, 1) AS B#,
        SUBSTR(DIGITS(PLANNO), 5, 1) AS P#,
        SUBSTR(DIGITS(METHOD), 5, 1) AS MTH,
        SUBSTR(TNAME, 1, 8) AS TBNAME,
        SUBSTR(DIGITS(TABNO), 5, 1) AS T#,
        SUBSTR(ACCESSTYPE, 1, 2) AS A,
        SUBSTR(DIGITS(MATCHCOLS), 5, 1) AS MC,
        SUBSTR(ACCESSNAME, 1, 8) AS IXNAME,
        SUBSTR(INDEXONLY, 1, 1) AS IO,
        SUBSTR(PREFETCH, 1, 1) AS PF,
        OPTHINT,
        HINT_USED,
        TSLOCKMODE,
        SUBSTR(TIMESTAMP, 7, 8)
FROM  PLAN_TABLE
ORDER BY 1, 2, 3
;
------------------------------------------------------------------------
Q#  B# P#  MTH  TBNAME    T#  A MC IXNAME    IO  PF OPTHINT HINTU
001 1  1   0    PRODUCTS  1   I 1  IND_PRDA  N   L  MYHINT
001 1  1   0    PRODUCTS  1   I 1  IND_PRDA  N   L          MYHINT
------------------------------------------------------------------------
-- COST ESTIMATION
SELECT  SUBSTR(DIGITS(QUERYNO), 7, 4) AS Q9,
        STMT_TYPE AS S_TYPE,
        COST_CATEGORY AS CC,
        SUBSTR(DIGITS(PROCSU), 1, 10) AS SU,
        REASON
FROM  DSN_STATEMNT_TABLE
ORDER BY 1, EXPLAIN_TIME
------------------------------------------------------------------------
Q9    S_TYPE  CC  SU          REASON
0001  SELECT  B   0000001527  HOST VARIABLES
0001  SELECT  B   0000001086  HOST VARIABLES
------------------------------------------------------------------------
```