ISO/IEC JTC 1/SC 32 N 1248

Date: 2005-04-01

REPLACES: --

CORRECTED

ISO/IEC JTC 1/SC 32

Data Management and Interchange

Secretariat: United States of America (ANSI)
Administered by Farance, Inc. on behalf of ANSI

DOCUMENT TYPE	Summary of Voting/Table of Replies
TITLE	Summary of Voting/Table of Replies for 32N1201 - ISO/IEC CD 9075-04 Information technology -- Database Languages - SQL - Part 4: Persistent Stored Modules (SQL/PSM)
SOURCE	SC 32 Secretariat
PROJECT NUMBER	1.32.03.06.04.00
STATUS	WG 3 should take and resolve the comments . Corrected 2005-04-01
REFERENCES	
ACTION ID.	ACT
REQUESTED ACTION	
DUE DATE	
Number of Pages	117
LANGUAGE USED	English
DISTRIBUTION	P & L Members SC Chair WG Conveners and Secretaries

Douglas Mann, Secretary, ISO/IEC JTC 1/SC 32
Farance, Inc *, 360 Pelissier Lake Road, Marquette, MI, United States of America
Telephone: +1 906-249-9275; Facsimile; E-mail: MannD@battelle.org
available from the JTC 1/SC 32 WebSite http://staging.jtc1sc32.org/
*Farance, Inc. administers the ISO/IEC JTC 1/SC 32 Secretariat on behalf of ANSI

ISO/IEC JTC 1/SC 32 N1248

Summary of Voting on Document SC 32 N 1202,

Title: ISO/IEC CD 9075-04 Information technology -- Database Languages - SQL - Part 4: Persistent Stored Modules (SQL/PSM)

"P" Member	Approval	Approval with Comments	Disapproval	Abstention
Australia			**X**	
Belgium				
Brazil				
Canada			**X**	
China	**X**			
Czech Republic	**X**			
Egypt				
Finland				
Germany		**X**		
Italy				**X**
Japan			**X**	
Korea, Republic of	**X**			
Netherlands, The			**X**	
Norway				
Portugal				
Sweden	X			
United Kingdom			**X**	
United States			**X**	
Total "P"	**4**	**1**	**6**	**1**
"O" Member				
Austria				
Denmark				
France				
Russian Federation				
Switzerland				
Total "O"				

ITALY

Lack of Experts

Template for comments and secretariat observations

Date: 2005-03-14	Document: **32N1202 9075-4 PSM Persistent Stored Modules**

1	2	(3)	4	5	(6)	(7)
MB[1]	**Clause No./ Subclause No./ Annex** (e.g. 3.1)	**Paragraph/ Figure/Table/ Note** (e.g. Table 1)	**Type of com-ment**[2]	**Comment (justification for change) by the MB**	**Proposed change by the MB**	**Secretariat observations** on each comment submitted
AU	N/a	N/a	ge	Address defects in Annex E the Defect Reports in the annex titled Defect reports not addressed in this edition of this part of ISO/IEC 9075 be addressed for the following: 32N1202 9075-4 PSM - Address defects in annex E		

1 **MB** = Member body (enter the ISO 3166 two-letter country code, e.g. CN for China; comments from the ISO/CS editing unit are identified by ******)

2 **Type of comment:** **ge** = general **te** = technical **ed** = editorial

NOTE Columns 1, 2, 4, 5 are compulsory.

National Body CAN Comments — 2005-02-03

32N1198, *ISO/IEC CD 9075-01 Information technology - Database Languages - SQL - Part 1: Framework (SQL/Framework)*
32N1199, *ISO/IEC CD 9075-02 Information technology - Database Languages - SQL - Part 2: Foundation (SQL/Foundation)*
32N1201, *ISO/IEC CD 9075-03 Information technology - Database Languages - SQL - Part 3: Call-Level Interface (SQL/CLI)*
32N1202, *ISO/IEC CD 9075-04 Information technology - Database Languages - SQL - Part 4: Persistent Stored Modules (SQL/PSM)*
32N1203, *ISO/IEC CD 9075-09 Information technology - Database Languages - SQL - Part 9: Management of External Data (SQL/MED)*
32N1204, *ISO/IEC CD 9075-10 Information technology - Database Languages - SQL - Part 10: Object language bindings (SQL/OLB)*
32N1205, *ISO/IEC CD 9075-11 Information technology - Database Languages - SQL - Part 11: Information and Definition Schemas (SQL/Schemata)*
32N1206, *ISO/IEC CD 9075-13 Information technology - Database Languages - SQL - Part 13: SQL Routines and Types Using the Java™ Programming Language(SQL/JRT)*

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
CD SQL/Framework						
	CAN-P01-001		1-Major Technical	*P01-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
CD SQL/Foundation						
	CAN-P02-001		1-Major Technical	*P02-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
CD SQL/CLI						
	CAN-P03-001		1-Major Technical	*P03-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
CD SQL/PSM						
	CAN-P04-001		1-Major Technical	*P04-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution None provided with comment.	
CD SQL/MED						
	CAN-P09-001		1-Major Technical	*P09-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
CD SQL/OLB						
	CAN-P10-001		1-Major Technical	*P10-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
CD SQL/Schemata						
	CAN-P11-001		1-Major Technical	*P11-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
CD SQL/JRT						
	CAN-P13-001		1-Major Technical	*P13-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	

ISO/IEC JTC1/SC32/WG3:TXL-034 – DIN NI-32 N 0642

Authoritative Version: Adobe Acrobat Portable Document Format (PDF)

ISO

International Organization for Standardization

DIN

Deutsches Institut für Normung

Din NI-32
Database
ISO/IEC JTC 1/SC 32
Data Management and Interchange
WG 3
Database Languages

Title: German Comments on SC32 N 1156: ISO/IEC CD 9075-1, 2, 3, 4, 9, 10, 11, 13:200x(E)

Status: Consolidated comments to assist with resolution of CD ballot comments

Author: Jörn Bartels (Germany)

National Body DEU Comments — 2005-03-08

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
ISO/IEC FCD 9075-01:200x(E) SQL/Framework						
1	DEU-P01-010		1-Major Technical	*P01-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
ISO/IEC FCD 9075-02:200x(E) SQL/Foundation						
2	DEU-P02-010		1-Major Technical	*P02-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
3	DEU-P02-020		1-Major Technical	*P02-07.6 Table reference*	Queries of the form SELECT … FROM <joined table> ….. Do not seem to be supported anymore. This is due to changes proposed in DRS-077. **Solution** None provided with comment.	
ISO/IEC FCD 9075-03:200x(E) SQL/CLI						
4	DEU-P03-010		1-Major Technical	*P03-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
ISO/IEC FCD 9075-04:200x(E) SQL/PSM						
5	DEU-P04-010		1-Major Technical	*P04-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
6	DEU-P11-020	DEU-P11-030	1-Major Technical	*P04-18.2 "MODULE_PRIVILEGES" Table*	The table MODULE_PRIVILEGES stores the privileges granted on a specific module. The same information could be stored in the table USAGE_PRIVILEGES. This would simplify the definition schema and standardise the way, how privileges are stored. **Solution** None provided with comment.	
7	DEU-P11-030	DEU-P11-020	1-Major Technical	*P04-18.2 "MODULE_PRIVILEGES" Table*	The constraint MODULE_PRIVILEGE_GRANTOR_CHECK and MODULE_PRIVILEGE_GRANTEE_CHECK reference still the tables ROLES and USERS. They are gone!	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution Replace the check constraints with a foreign key on AUTHORIZATIONS.	
ISO/IEC FCD 9075-09:200x(E) SQL/MED						
8	DEU-P09-010		1-Major Technical	*P09-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
9	DEU-P09-020		2-Minor Technical	*P09-25 Definition Schema*	Some tables which are introduced by this clause have for most columns no NOT NULL constraint, where it should be defined. This are at least 25.4 FOREIGN_DATA_WRAPPERS base table, 25.8 FOREIGN_TABLES base table and 25.10 ROUTINE_MAPPINGS base table **Solution** None provided with comment.	
10	DEU-P09-030		3-Major Editorial	*P09-25.2 DATA_TYPE_DESCRIPTOR" table*	The constraint DATA_TYPE_DESCRIPTOR_DATA_TYPE_CHECK_COMBINATIONS is entirely replaced. This leads to problems of desynchronisation with SQL/Schemata. It does also not allow modifications from other parts (like SQL/XML) of the standard. The constraint does also currently not check the NULL applicability of the columns, as described in Description 2), which are inserted by this constraint. **Solution** None provided with comment.	
11	DEU-P09-040		3-Major Editorial	*P09-25.2 DATA_TYPE_DESCRIPTOR" table*	The Descriptions 2) and 3) are in conflict with each other. They describe both the NULLability of the newly introduced columns. There is a conflict if both come to different results. **Solution** The Descriptions 2) and 3) should be merged.	
12	DEU-P09-050		2-Minor Technical	*P09-25.4 "FOREIGN_DATA_WRAPPERS" table*	There is no constraint, which verifies the existence of the catalog and the authorization Identifier, which is used. **Solution** None provided with comment.	
13	DEU-P09-060		2-Minor Technical	*P09-25.6 "FOREIGN_SERVERS" table*	There is no constraint, which verifies the existence of the catalog and the authorization Identifier, which is used. **Solution** None provided with comment.	
14	DEU-P09-070		2-Minor Technical	*P09-25.12 "TABLES" table*	There is no constraint, which verifies that for a FOREIGN table there is also an entry in the table FOREIGN_TABLES. This could be done as it is done already in constraint TABLES_CHECK_NOT_VIEW of the table TABLES.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution None provided with comment.	
15	DEU-P09-080		2-Minor Technical	*P09-25.13 "USAGE_PRIVILEGES" table*	There is no constraint added, which checks the values of OBJECT_CATALOG and OBJECT_NAME as described in Description 1) There is also no modification of the constraint USAGE_PRIVILEGES_CHECK_REFERENCES_OBJECT which allows OBJECT_SCHEMA to be the empty string. It is currently also not allowed, that the OBJECT_TYPE is anything except 'DOMAIN', 'CHARACTER SET', 'COLLATION', 'TRANSLATION', 'SEQUENCE'. The use for a foreign-data wrapper or a foreign server requires a modification of constraint USAGE_PRIVILEGES_OBJECT_TYPE_CHECK. **Solution** None provided with comment.	
16	DEU-P09-090		2-Minor Technical	*P09-25.15 "USER_MAPPINGS" table*	There is no foreign key check for the column AUTHORIZATION_IDENTIFIER. **Solution** None provided with comment.	
ISO/IEC FCD 9075-10:200x(E) SQL/OLB						
17	DEU-P10-010		1-Major Technical	*P10-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
ISO/IEC FCD 9075-11:200x(E) SQL/Schemata						
18	DEU-P11-010		1-Major Technical	*P11-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	
19	DEU-P11-020		1-Major Technical	*P11-No specific location*	It is not clear, which tables should have a direct or indirect relationship to the table SCHEMATA. For some tables is a foreign key defined, for some is a check constraint defined, which checks the foreign key relationship only when there are schemas in the same catalog. There should be an explanation for this distinction and all relationships need to be checked for correctness. An example for a dubious relationship is the constraint TRIGGERS_REFERENCES_TABLES. Is it really possible, to define a trigger on a table of another catalog. This is especially strange, as the table	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					TRIGGERED_UPDATE_COLUMNS has a direct foreign key to COLUMNS. **Solution** None provided with comment.	
20	DEU-P11-030		3-Major Editorial	*P11-5.54 Short name views*	The View definitions in 5.54 Short name views should be sorted according to the order of the base views (i.E. position of CONSTR_COL_USAGE). **Solution** Order them according to base view order.	
21	DEU-P11-040		2-Minor Technical	*P11-5.78 "SQL_LANGUAGES" View*	The View SQL_LANGUAGES is depricated. In Note 6 is a reference to SQL/Framework Subclause 6.4. There is a backwards reference, in Note 9, which says "The equivalent information is available to the SQL user in the Information Schema." When we delete the view, it is not clear if the Note 9 should also be deleted. **Solution** None provided with comment.	
22	DEU-P11-050		2-Minor Technical	*P11-6.9 "CHARACTER_ENCODING_FORMS" Table*	The NOT NULL Constraints are not needed, as all columns are part of the primary key. **Solution** Delete the NOT NULL constraints.	
23	DEU-P11-060		2-Minor Technical	*P11-6.10 "CHARACTER_REPERTOIRES" Table*	The NOT NULL Constraints on the column CHARACTER_REPERTOIRE_NAME is not needed, as the column is part of the primary key. **Solution** Delete the NOT NULL constraint.	
24	DEU-P11-070		4-Minor Editorial	*P11-6.11 "CHARACTER_SETS" Table*	The column NUMBER_OF_CHARACTERS is in the last Edition of the Standard depricated and should now be deleted. **Solution** Delete the column. Do the same in the View Definition 5.12 CHARACTER_SETS view and in 5.78 Short name views in the view CHARACTER_SETS_S. Delete the corresponding List Elements 1) and 2) in Annex C.	
25	DEU-P11-080		2-Minor Technical	*P11-6.16 "COLLATIONS" Table*	There is no constraint for the column CHARACTER_REPERTOIRE_NAME defined. It needs to reference the Table CHARACTER_REPERTOIRES. **Solution** Add the constraint COLLATIONS_FOREIGN_KEY_CHARACTER_REPERTOIRES FOREIGN KEY (CHARACTER_REPERTOIRES) REFERENCES CHARACTER_REPERTOIRES.	
26	DEU-P11-090		4-Minor Editorial	*P11-6.16 "COLLATIONS" Table*	The columns COLLATION_TYPE, COLLATION_DICTIONARY, and COLLATION_DEFINITION are in the last edition of the Standard depricated and should now be deleted. **Solution** Delete the columns. Do the same in the View Definition 5.15 COLLATIONS view and in 5.78 Short name views in the view COLLATIONS_S. Delete the corresponding List Elements 3) and 4) in Annex C.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
27	DEU-P11-100		2-Minor Technical	*P11-6.16 "COLLATIONS" Table*	The column PAD_ATTRIBUTE has no NOT NULL check constraint, even that there is in the description no explanation of the meaning of a possible NULL value. **Solution** None provided with comment.	
28	DEU-P11-110		2-Minor Technical	*P11-6.20 "COLUMNS" Table*	For the columns IS_GENERATED and IDENTITY_GENERATION are no check constraints specified, but in the description are Lists of allowed values. **Solution** Add to the column IS_GENERATED the following column level constraint: CONSTRAINT COLUMNS_IS_GENERATED_CHECK CHECK (IS_GENERATED in ('NEVER', 'ALWAYS')) Add to the column IDENTITY_GENERATION the following column level constraint: CONSTRAINT COLUMNS_ IDENTITY_GENERATION_CHECK CHECK (IDENTITY_GENERATION IN ('ALWAYS', 'BY DEFAULT'))	
29	DEU-P11-120		2-Minor Technical	*P11-6.21 "DATA_TYPE_DESCRIPTORS" Table*	The constraint DATA_TYPE_DESCRIPTOR_FOREIGN_KEY_SCHEMATA assures that the values of USER_DEFINED_TYPE_CATALOG and USER_DEFINED_TYPE_SCHEMA have corresponding rows in the table SCHEMATA. The constraint DATA_TYPE_DESCRIPTOR_CHECK_REFERENCES_UDT allows that the value for the column USER_DEFINED_TYPE_CATALOG has no corresponding row in SCHEMATA. As this is not possible according to the first constraint, we could rewrite this constraint as a foreign key. It is not clear, if this is intended. **Solution** None provided with comment.	
30	DEU-P11-130		2-Minor Technical	*P11-6.21 "DATA_TYPE_DESCRIPTORS" Table*	The columns SCOPE_CATALOG, SCOPE_SCHEMA, and SCOPE_NAME are not checked against the possible values in the table TABLES. **Solution** None provided with comment.	
31	DEU-P11-140		4-Minor Editorial	*P11-6.21 "DATA_TYPE_DESCRIPTORS" Table*	The constraint DATA_TYPE_DESCRIPTOR_CHECK_OBJECT_TYPE should be a column constraint, as it references only the column OBJECT_TYPE. **Solution** Remove the preceding comma.	
32	DEU-P11-150		4-Minor Editorial	*P11-6.21 "DATA_TYPE_DESCRIPTORS" Table*	In the constraint DATA_TYPE_DESCRIPTOR_CHECK_REFERENCES_COLLATION_CHARACTER_SET_APPLICABILITY should be a comma "," at the end of the 10th line of the constraint. **Solution** Add the missing comma.	
33	DEU-P11-		2-Minor Technical	*P11-6.24 "DOMAIN_CONSTRAINTS" Table*	Should there be a NOT NULL check constraint for the columns IS_DEFERRABLE and INITIALLY_DEFERRED?	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
	160				In the description is no explanation of the meaning of a possible null value. **Solution** None provided with comment.	
34	DEU-P11-170		2-Minor Technical	*P11-6.31 "PARAMETERS" Table*	The foreign key constraint PARAMETERS_FOREIGN_KEY_SCHEMATA does not check the name of the routine. This constraint should be removed and instead there should be a foreign key to ROUTINES be defined. **Solution** None provided with comment.	
35	DEU-P11-180		2-Minor Technical	*P11-6.31 "PARAMETERS" Table*	There needs to be a unique constraint defined, which guarantees the uniqueness of a parameter name for a routine. **Solution** Add the constraint: CONSTRAINT PARAMETERS_UNIQUE_CHECK (UNIQUE SPECIFIC_CATALOG, SPECIFIC_SCHEMA, SPECIFIC_NAME, PARAMETER_NAME)	
36	DEU-P11-190		2-Minor Technical	*P11-6.31 "PARAMETERS" Table*	For the columns FROM_SQL_SPECIFIC_CATALOG, FROM_SQL_SPECIFIC_SCHEMA, and FROM_SQL_SPECIFIC_ NAME and TO_SQL_SPECIFIC_CATALOG, TO_SQL_SPECIFIC_SCHEMA, and TO_SQL_SPECIFIC_NAME is no foreign key check defined. **Solution** None provided with comment.	
37	DEU-P11-200		1-Major Technical	*P11-6.36 "ROUTINE_PRIVILEGES" Table*	The table ROUTINE_PRIVILEGES stores the privileges granted on a specific routine. The same information could be stored in the table USAGE_PRIVILEGES. This would simplify the definition schema and standardise the way, how privileges are stored. **Solution** None provided with comment.	
38	DEU-P11-210		2-Minor Technical	*P11-6.41 "SCHEMATA" Table*	For the columns DEFAULT_CHARACTER_SET_CATALOG, DEFAULT_CHARACTER_SET_SCHEMA and DEFAULT_CHARACTER_SET_NAME is a foreign key referencing the table CHARACTER_SETS missing. **Solution** Add the missing Foreign Key constraint: CONSTRAINT SCHEMATA_FOREIGN_KEY_ CHARACTER_SETS FOREIGN KEY (DEFAULT_CHARACTER_SET_CATALOG, DEFAULT_CHARACTER_SET_SCHEMA, DEFAULT_CHARACTER_SET_NAME) REFERENCES CHARACTER_SETS	
39	DEU-P11-		4-Minor Editorial	*P11-6.48 "TABLE_CONSTRAINTS" Table*	The constraint TABLE_CONSTRAINTS_UNIQUE_CHECK is not needed, as the uniqueness of the constraint name is already checked by the assertion UNIQUE_CONSTRAINT_NAME in subclause 6.4	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
	220				**Solution** A possible solution is to remove the superflouus constraint.	
40	DEU-P11-230		4-Minor Editorial	*P11-6.50 "TABLE_PRIVILEGES" Table*	In the constraint TABLE_PRIVILEGES_TYPE_CHECK is the last element of the inlist misspelled. It should be REFERENCES instead of EFERENCES **Solution** Fix the typo.	
41	DEU-P11-240		4-Minor Editorial	*P11-6.54 "TRIGGERED_UPDATE_COLUMNS" Table*	The constraint TRIGGERED_UPDATE_COLUMNS_FOREIGN_KEY_TRIGGERS is not needed, as a more restrictive relationship is already guaranteed by constraint TRIGGERED_UPDATE_COLUMNS_EVENT_MANIPULATION_CHECK. **Solution** A possible solution is to remove the superflouus constraint.	
42	DEU-P11-250		2-Minor Technical	*P11-6.55 "TRIGGER_COLUMN_USAGE" Table*	The table TRIGGER_COLUMN_USAGE should have a foreign Key to the table TRIGGER_TABLE_USAGE, and not to TRIGGERS. **Solution** Add the following constraint: TRIGGER_COLUMN_USAGE_FOREIGN_KEY_TRIGGER_TABLE_USAGE FOREIGN KEY (TABLE_CATALOG, TABLE_SCHEMA, TABLE_NAME) REFERENCES TRIGGER_TABLE_USAGE It might be possible to remove the constraint TRIGGER_COLUMN_USAGE_FOREIGN_KEY_TRIGGERS.	
43	DEU-P11-260		2-Minor Technical	*P11-6.62 "USER_DEFINED_TYPES" Table*	In the last query of the constraint USER_DEFINED_TYPES_CHECK_SOURCE_TYPE is the column OBJECT_TYPE not in the reference List of the IN clause. **Solution** None provided with comment.	
44	DEU-P11-270		2-Minor Technical	*P11-Appendix C 6)*	The columns FEATURE_ID and FEATURE_NAME of the view SQL_PACKAGES are in the last Edition of the Standard depricated and should now be deleted. But without these columns does the view not provide any usefull information. Should the entire View be deleted? **Solution** None provided with comment.	
ISO/IEC FCD 9075-13:200x(E) SQL/JRT						
45	DEU-P13-010		1-Major Technical	*P13-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	

END OF PAPER

ISO/IEC JTC1/SC32/WG3 TXL-nnn

3 March, 2005

ISO

International Organization for Standardization

ISO/IEC JTC 1/SC 32
Data Management and Interchange
WG 3
Database Languages

Title: Japan Ballot Comments on CD 9075:2007(E)

Status: Document to accompany ballot response

Author: Masashi Tsuchida ,Takaaki Shiratori, Takashi Kotera

Abstract: We present the comments of Japan on the CD ballot documents, to accompany our ballot response on that document.

References:
[1] WG3:TXL-002 = 32N1198, ISO/IEC CD 9075-1, Information technology . Database languages SQL . Part 1: Framework (SQL/Framework)]
[2] WG3:TXL-003 = 32N1199, ISO/IEC CD 9075-2, Information technology . Database languages . SQL . Part 2: Foundation (SQL/Foundation)
[3] WG3:TXL-004 = 32N1201, ISO/IEC CD 9075-3, Information technology . Database languages . SQL . Part 3: Call-Level Interface (SQL/CLI)
[4] WG3:TXL-005 = 32N1202, ISO/IEC CD 9075-4, Information technology . Database languages . SQL . Part 4: Persistent Stored Modules (SQL/PSM)
[5] WG3:TXL-006 = 32N1203, ISO/IEC CD 9075-9, Information technology . Database languages . SQL . Part 9: Management of

External Data (SQL/MED)
[6] WG3:TXL-007 = 32N1204, ISO/IEC CD 9075-10, Information technology . Database languages . SQL . Part 10: Object language bindings (SQL/OLB)
[7] WG3:TXL-008 = 32N1205, ISO/IEC CD 9075-11, Information technology . Database languages . SQL . Part 11: Information and Definition Schemas (SQL/Schemata)
[8] WG3:TXL-009 = 32N1206, ISO/IEC CD 9075-11, Information technology . Database languages . SQL . Part 13: Information and Definition Schemas (SQL/JRT)

ISO/IEC JTC1/SC32/WG3 TXL-nnn

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
SQL/Framework						
001	JPN-P01-001		1-Major Technical	*P01-No specific location*	There are quite a few features to discuss a emerging next standard. Japan thinks that we should take enough time to add new features. **Solution** None provided with comment.	
SQL/Foundation						
002	JPN-P02-002		1-Major Technical	*P02-No specific location*	There are quite a few features to discuss a emerging next standard. Japan thinks that we should take enough time to add new features. **Solution** None provided with comment.	
003	JPN-P02-003		1-Major Technical	*P02-11.3, <table definition>*	It is allowed that <table contents source> which is <as subquery clause> with WITH DATA is specified for a temporary table. But a temporary table can not be materialized at table definition. **Solution** None provided with comment.	
004	JPN-P02-004		1-Major Technical	*P02-14.8, <insert statement>*	An INSERT statement has no different effects on identity columns specified GENERATED ALWAYS and that specified GENERATED BY DEFAULT. **Solution** None provided with comment.	
SQL/CLI						
005	JPN-P03-001		1-Major Technical	*P03-No specific location*	There are quite a few features to discuss a emerging next standard. Japan thinks that we should take enough time to add new features. **Solution** None provided with comment.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
SQL/PSM						
006	JPN-P04-001		1-Major Technical	*P04-No specific location*	There are quite a few features to discuss a emerging next standard. Japan thinks that we should take enough time to add new features. **Solution** None provided with comment.	
SQL/MED						
007	JPN-P05-001		1-Major Technical	*P05-No specific location*	There are quite a few features to discuss a emerging next standard. Japan thinks that we should take enough time to add new features. **Solution** None provided with comment.	
SQL/OLB						
008	JPN-P10-001		1-Major Technical	*P10-No specific location*	There are quite a few features to discuss a emerging next standard. Japan thinks that we should take enough time to add new features. **Solution** None provided with comment.	
SQL/Schema						
009	JPN-P11-001		1-Major Technical	*P11-No specific location*	There are quite a few features to discuss a emerging next standard. Japan thinks that we should take enough time to add new features. **Solution** None provided with comment.	
SQL/JRT						
009	JPN-P13-001		1-Major Technical	*P13-No specific location*	There are quite a few features to discuss a emerging next standard. Japan thinks that we should take enough time to add new features. **Solution** None provided with comment.	

ISO/IEC JTC1/SC32/WG3 TXL-031
2004-01-23

ISO
International Organization for Standardization
ISO/IEC JTC 1/SC 32
Data Management and Interchange
WG 3
Database Languages

Project: ISO: 1.32.3.5

Title: Ballot Comment on ISO/IEC CD 9075-1, -2, -3, -4, -9, -10, -11, and -13

Status: Netherlands National Body Comments

Author: Stephen Cannan (Editor)

References:

[1] SC32 N01198, CD 9075-1 Information Technology - Database Language SQL - Part 1: Framework (SQL/Framework) Jim Melton (Editor), December, 2004.
[2] SC32 N01199, CD 9075-2 Information Technology - Database Language SQL - Part 2: Foundation (SQL/Foundation) Jim Melton (Editor), December, 2004.
[3] SC32 N01201, CD 9075-3 Information Technology - Database Language SQL - Part 3: Call Level Interface (SQL/CLI) Jim Melton (Editor), December, 2004.
[4] SC32 N01202, CD 9075-4 Information Technology - Database Language SQL - Part 4: Persistent Stored Modules (SQL/PSM) Jim Melton (Editor), December, 2004.
[5] SC32 N01203, CD 9075-9 Information Technology - Database Language SQL - Part 9: Management of External Data (SQL/MED) Jim Melton (Editor), December, 2004.
[6] SC32 N01204, CD 9075-10 Information Technology - Database Language SQL - Part 10: Object Language Bindings (SQL/OLB) Jim Melton (Editor), December, 2004.
[7] SC32 N01205, CD 9075-11 Information Technology - Database Language SQL - Part 11: Schemata (SQL/Schemata) Jim Melton (Editor), December, 2004.
[8] SC32 N01206, CD 9075-13 Information Technology - Database Language SQL - Part 13: Java Routines and Types (SQL/JRT) Jim Melton (Editor), December, 2004.

The Netherlands vote is:

SQL/Framework	No with comments
SQL/Foundation	No with comments
SQL/CLI	No with comments
SQL/PSM	No with comments
SQL/MED	No with comments
SQL/OLB	No with comments
SQL/Schemata	No with comments
SQL/JRT	Yes with comments

If all problems and technical errors, i.e. those identified in this ballot, and those identified during the editing meeting(s), are resolved to our satisfaction, then the Netherlands will change its NO votes to YES votes.

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
SQL/Framework						
	NLD-P01-001		3-Major Editorial	*P01-04.04, SQL data types*	FRM-002 The following Language Opportunity has been noted: **Source:** DBL:BBN-167/X3H2-98-386 **Language Opportunity:** Section needs a better organization There should be a section called SQL Data Types. Then a short definition of what is meant by an SQL data type. Then the list of the five types of data types (predefined, row type, user-defined type, collection type, and reference type). Then there should be a definition for each. **Solution** None provided with comment.	
	NLD-P01-002		2-Minor Technical	*P01-06.03.03.03, Rule evaluation order*	**WG3-P01-001** The referenced subclause includes the following text: In general, if some syntactic element contains more than one other syntactic element, then the General Rules for contained elements that appear earlier in the production for the containing syntactic element are applied before the General Rules for contained elements that appear later. For example, in the production: `<A> ::= <B> <C>` the Syntax Rules and Access Rules for <A>,<B>,and <C>are effectively applied simultaneously. The General Rules for <B>are applied before the General Rules for <C>, and the General Rules for <A>are applied after the General Rules for both <B>and <C>. In SQL/Foundation, Subclause 13.5, "<SQL procedure statement>", is a clear exception to this general rule for General Rules, for the GRs of the particular contained statement (e.g., an <insert statement>) are clearly intended to be invoked only when a GR in Subclause 13.5 explicitly states that the contained statement to be executed. Now, it might be that the introductory words, "In general", can be taken to imply that there are some exceptions, but in that case shouldn't the exceptions be explicitly mentioned? **Solution** None provided with comment.	
SQL/Foundation						
	NLD-P02-001		1-Major Technical	*P02-04.32.01, General description of cursors*	FND-975 The following Possible Problem has been noted: **Source:** WG3:SIA-030 = H2-2004-??? **Possible Problem:** Subclause 4.32.1, “General description of cursors”, contains: For every <declare cursor> **in** [emphasis added] an SQL-client module, a cursor	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					is effectively created when an SQL-transaction (see Subclause 4.35, “SQL-transactions”) referencing the SQL-client module is initiated. For every <dynamic declare cursor> in an <SQL-client module definition>, a cursor is effectively created when an SQL-transaction (see Subclause 4.35, “SQL-transactions”) referencing the <SQLclient module definition> is initiated. An extended dynamic cursor is **also** [emphasis added] effectively created when an <allocate cursor statement> is executed within an SQL-session and destroyed when that SQL-session is terminated. This text suffers from several problems, all of which probably need to be addressed at the same time: 1) The first paragraph entertains the notion of a piece of SQL syntax appearing inside something that is not a piece of SQL syntax. It seems that either "<declare cursor>" should be replaced by "cursor", or "SQL-client module" should be replaced by "<SQL-client module definition>". In either case there would be knock-on effects on the remaining text. Note that the second paragraph prefers to talk about syntactic containment exclusively, but its text is too suspect for it to be used as a guideline for correcting the first paragraph. 2) The first paragraph entertains the notion of an SQL-transaction referencing an SQL-client module. Regardless of whether this should be SQL-client module or <SQL-client module definition>, it is not clear exactly what it means for an SQL-transaction that is the process of being initiated to “reference” that thing. Text elsewhere in SQL/Foundation (for example, in Subclause 16.7, “<commit statement>”), strongly suggests that several distinct SQL-client modules can be “associated with” the same current SQL-transaction. Can they be associated with the SQL-transaction without also being referenced by it? For that matter, can they be referenced by it without also being associated with it? If “referenced by” and “associated with” are synonymous, then how can all the SQL-client modules referenced by SQL-transaction *T* be known when *T* is initiated? 3) The second paragraph entertains the notion of creation of a cursor, and yet we have not been able to find any mention of this concept in any General Rule. Subclause 19.8, “<deallocate prepared statement>”, GR3) does require destruction of certain cursors, and this is corroborated (redundantly?) by Subclause 19.15, “<allocate cursor statement>”, GR3)d). However, neither Subclause 19.6, “<prepare statement>”, nor Subclause 19.15, “<allocate cursor statement>”, has any GR requiring a cursor to be created. 4) As already noted, a cursor, having been created in somewhat mysterious circumstances, is never destroyed (unless it happens to be an allocated cursor, or a cursor declared in a <compound statement>, see SQL/PSM, Subclause 13.1, “<compound statement>”, GR3)c)ii)2) and GR5)). It seems, then, that if *n* SQL-transactions in the same SQL-session “reference” the same SQL-client module,	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					then each <declare cursor> contained in the corresponding <SQL-client module definition> causes the creation of *n* distinct cursors. And yet Subclause 14.2, "<open statement>", SR1), says "Let *CR* be the cursor specified by *DC*", where *DC* is a <declare cursor>. There are two problems with this: — It is not clear which of those *n* cursors is the one specified by *DC*. Of course, if the standard clearly specified that all but one of these had been destroyed by this time, then there would be no ambiguity. — The cited text in Subclause 4.32.1, "General description of cursors", makes it clear that a cursor comes into existence at run-time and therefore, not being a schema object, should not be referred to in a syntax rule. Since the SQL-session context already includes cursor positions, perhaps it should also be defined to include cursors. 5) The final sentence contains the word "also", which could be understood as suggesting that some way of creating an extended dynamic cursor has already been mentioned. Moreover, it ignores the possibility of such a cursor being destroyed as a consequence of its prepared statement being deallocated before SQL-session termination. **Solution** None provided with comment.	
	NLD-P02-002		2-Minor Technical	*P02-03.01, Definitions*	FND-953 The following Language Opportunity has been noted: **Language Opportunity:** There has been a discussion about Unicode 4.0 on the ISODBL list. [Ake has] found out that Note 7 in SQL/Foundation will be affected, bescuase it contains explicit code points. U+180E and U+205F have been added to the "Zs" class in Unicode 4.0. Note that U+200B currently is of class "Zs", although it should not be treated as white-space. The Unicode Technical Committee will probably change the class for U+200B (ZERO-WIDTH SPACE) to "Cf" in the near future. **Solution** None provided with comment.	
	NLD-P02-003		2-Minor Technical	*P02-04.10, Collection types*	FND-845 The following Language Opportunity has been noted: **Source:** WG3:YYJ-016 (CAN-P02-001, USA-P02-005) **Language Opportunity:** The next edition of the SQL standard should standardize the syntax and semantics of one or more additional collection types. **Solution** None provided with comment.	
	NLD-P02-004		1-Major Technical	*P02-04.14.02, Types of tables*	FND-944 The following Possible Problem has been noted: **Source:** WG3:HBA-042 = H2-2003-___ **Possible Problem:** This Subclause, with paragraph numbers added for expository purposes, says of created temporary tables:	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					1) A global temporary table is a named table defined by a <table definition> that specifies GLOBAL TEMPORARY. A created local temporary table is a named table defined by a <table definition> that specifies LOCAL TEMPORARY. Global and created local temporary tables are effectively materialized only when referenced in an SQL-session. Every SQL-client module in every SQL-session that references a created local temporary table causes a distinct instance of that created local temporary table to be materialized. That is, the contents of a global temporary table or a created local temporary table cannot be shared between SQL-sessions. 2) In addition, the contents of a created local temporary table cannot be shared between SQL-client modules of a single SQL-session. The definition of a global temporary table or a created local temporary table appears in a schema. In SQL language, the name and the scope of the name of a global temporary table or a created local temporary table are indistinguishable from those of a persistent base table. However, because global temporary table contents are distinct within SQL-sessions, and created local temporary tables are distinct within SQL-client modules within SQL-sessions, the *effective* <schema name> of the schema in which the global temporary table or the created local temporary table is instantiated is an implementation-dependent <schema name> that may be thought of as having been effectively derived from the <schema name> of the schema in which the global temporary table or created local temporary table is defined and the implementation-dependent SQLsession identifier associated with the SQL-session. 3) In addition, the *effective* <schema name> of the schema in which the created local temporary table is instantiated may be thought of as being further qualified by a unique implementation-dependent name associated with the SQL-client module in which the created local temporary table is referenced. 4) A declared local temporary table is a module local temporary table. A module local temporary table is a named table defined by a <temporary table declaration> in an SQL-client module. A module local temporary table is effectively materialized the first time it is referenced in an SQL-session, and it persists for that SQL-session. Neither the first sentence of paragraph 1 nor the General Rules of Subclause 11.3, "<table definition>", make it clear that a <table definition> creates a persistent (temporary) table descriptor. *Materialised* is not defined and the meaning added by the qualifier *effectively* is unclear; the use of *instantiated* in paragraph 3 suggests a distinction that is probably unintended. In paragraph 2, "*effective* <schema name> ... may be thought of as ..." [emphasis in original] doesn't tell us what the purpose of this thinking is, nor how the *effective* <schema name> differs from any possible *actual* one. It also misleads us into imagining that a local temporary table created in the same schema as a	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					global one might have the same <table name>, when on reflection it clearly can't. This is not, be it noted, analogous to extended names, as explicitly intended by DBL:LON-156. Furthermore, “unique implementation-dependent name associated with the SQL-client module in which the created local temporary table is referenced” is unclear. Is this name persistent? One interpretation is that every occurrence of the <table name> must be contained in the same <SQL-client module definition>. But perhaps it is intended to mean that there is no restriction on where the <table name> can occur; but only occurrences in externally-invoked procedures in the same module refer to the same thing (i.e. those of other modules refer to their own “local” temporary table). In paragraph 2, it is not clear whether a distinction is intended between “global temporary table contents” and “created local temporary tables”, but presumably not. In paragraph 2, the meaning of “distinct within SQL-sessions” is unclear, because there is only one SQLsession active at any one time (even though there may be dormant ones). **Solution** None provided with comment.	
	NLD-P02-005		1-Major Technical	*P02-04.14.02, Types of tables*	FND-945 The following Possible Problem has been noted: **Source:** WG3:HBA-042 = H2-2003-___ **Possible Problem:** In this Subclause, as modified by [PSM-WD], Subclause 4.3.1, “Types of tables”, with paragraph numbers added for expository purposes, says of declared temporary tables: 1) A declared local temporary table is a module local temporary table. A module local temporary table is a named table defined by a <temporary table declaration> in an SQL-client module. A module local temporary table is effectively materialized the first time it is referenced in an SQL-session, and it persists for that SQL-session. 2) A declared local temporary table may be declared in an SQL-client module. 3) Inserted by SQL/PSM A declared local temporary table may be declared in an SQL-server module. 4) A declared local temporary table that is declared in an SQL-client module is a named table defined by a <temporary table declaration> that is effectively materialized the first time any <externallyinvoked procedure> in the <SQL-client module definition> that contains the <temporary table declaration> is executed. A declared local temporary table is accessible only by <externally-invoked procedure>s in the <SQL-client module definition> that contains the <temporary table declaration>. The effective <schema name> of the <schema qualified name> of the declared local temporary table may be thought of as the	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					implementation-dependent SQL-session identifier associated with the SQL-session and a unique implementation-dependent name associated with the <SQL-client module definition> that contains the <temporary table declaration>. 5) Inserted by SQL/PSM A declared local temporary table that is declared in an SQL-server module is a named table defined by a <temporary table declaration> that is effectively materialized the first time any <module routine> in the <SQL-server module definition> that contains the <temporary table declaration> is executed. A declared local temporary table is accessible only by <module routine>s in the <SQL-server module definition> that contains the <temporary table declaration>. The effective <schema name> of the <schema qualified name> of the declared local temporary table may be thought of as the implementation-dependent SQL-session identifier associated with the SQL-session and the name of the <SQL-server module definition> that contains the <temporary table declaration>. The second sentence of paragraph 1 is no longer true when paragraphs 3 and 5 have been inserted by PSM. Moreover, whatever truth is expressed by paragraph 1 is repeated by paragraphs 2 and 4, which are specific to SQL-client modules. Evidently paragraphs 2 and 4 were inserted to correspond to paragraphs 3 and 5, so making paragraph 1 redundant, which should have been deleted at the same time. To say, in paragraphs 4 and 5, that a declared local temporary table has an *effective* <schema name> is misleading, since its name must be prefixed by MODULE. **Solution** None provided with comment.	
	NLD-P02-006		2-Minor Technical	*P02-04.14.02, Types of tables*	FND-969 The following Language Opportunity has been noted: **Source:** WG3:SIA-018 = H2-2004-429 **Language Opportunity:** Every view component is an underlying table. The reason that underlying table terminology was not used was that the hierarchy of underlying tables does not follow the hierarchy of syntactic containment, owing to the distinctive treatment accorded the tables and derived tables in the FROM clause of a <query specification> compared with other derived tables found in a <query specification>. If this issue can be overcome, it may be possible to eliminate the notion of view component and just use underlying tables. **Solution** None provided with comment.	
	NLD-P02-007		1-Major Technical	*P02-04.17, Integrity constraints*	FND-703 The following Possible Problem has been noted: **Source:** WG3:BBN-139/X3H2-98-363 **Possible Problem:** It seems that SQL3's specification of deferrable constraints is ill-specified. Referential constraints are based on the notion of marking rows for deletion	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					before the rows are effectively deleted at the end of the SQL statement. This is necessary because updates cascaded by referential constraints need to be “propagated” through rows marked for deletion in order to avoid anomalies (non deterministic behavior). If a referential constraint is deferred, then rows that need to be kept around for the execution of referential constraints will not be present at the end of the transaction (or when the referential constraint is turned to immediate). These rows will be deleted at the end of the SQL statements. So, it is unclear how referential constraints are checked in these cases (e.g., are we supposed to maintain multiple versions of the database and check the constraints against those versions? If so, how do the updates are "propagated" to the current version of the database?). Another problem with deferrable constraints is that stored procedures and triggers can never rely on the existence of a consistent database during their execution because the application that caused the invocation of the stored procedure and/or trigger could have deferred the checking of certain constraints prior to the invocation of the procedure or trigger. (Please note that this has also a major impact to the implementation of such concepts because plans generated by optimizers (e.g., the exploitation of a unique index) can be invalidated by deferring such constraints.) Also it is not clear to me that deferrable constraints and triggers work smoothly. First, BEFORE triggers execute BEFORE the SQL statement that activates them. However, the BEFORE execution cannot be guaranteed if referential constraints are deferred because the execution of the BEFORE trigger needs to be deferred as well. Second, if the BEFORE trigger is modifying the values of transition variables such that they can be inserted/updated with correct values in the database, what will happen with such values if the BEFORE trigger executes after the database has been updated? Third, triggers are executed in a well defined order. This is important to guarantee that changes to the database are done in a deterministic manner. If constraints are deferred, then one may end up deferring the execution of several instances of the same trigger for which there is no well defined order of execution. This will lead to non-deterministic behavior in the database. **Solution** None provided with comment.	
	NLD-P02-008		2-Minor Technical	*P02-04.27, SQL-invoked routines*	FND-725 The following Language Opportunity has been noted: **Source:** WG3:FRA-122/X3H2-98-688) **Language Opportunity:** Subclause 4.27, “SQL-invoked routines”, does not adequately describe the concepts of dynamic binding and subject function selection. **Solution** None provided with comment.	
	NLD-P02-009		2-Minor	*P02-04.32,*	FND-607 The following Language Opportunity has been noted:	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
			Technical	*Cursors*	**Source:** DBL:LGW-146/X3H2-97-349 **Language Opportunity:** The ability to hold a cursor through rollback will be extremely useful to applications. Yet the second bullet of this Subclause says "a holdable-cursor is closed no matter what its state if the SQL-transaction is terminated with a rollback operation." This provision is not always necessary according to Jim Gray and Andrewas Reuter "Transaction Processing: Concepts and Techniques". **Solution** None provided with comment.	
	NLD-P02-010		2-Minor Technical	*P02-04.32.01, General description of cursors*	FND-929 The following Possible Problem has been noted: **Source:** WG3:HBA-040 **Possible Problem:** Although the second paragraph of this subclause defines terms to denote both varieties of dynamic cursors, it does not provide a way of referring to a cursor that is *not* dynamic. **Solution** None provided with comment.	
	NLD-P02-011		1-Major Technical	*P02-04.33.04, SQL-statements and transaction states*	FND-923 The following Possible Problem has been noted: **Source:** WG3:HBA-029 **Possible Problem:** Subclause 4.33.4, “SQL-statements and transaction states”, includes: If the initiation of an SQL-transaction occurs in an atomic execution context, and an SQL-transaction has already completed in this context, then an exception condition is raised: *invalid transaction termination.* At first sight it doesn't seem possible for transaction termination to be followed by transaction initiation "in" the same atomic execution context. In general, transaction initiation is caused by execution of an SQL-statement of the transaction-initiating kind and transaction termination is caused by executing an SQL-statement of a different kind (COMMIT or ROLLBACK). Note that Subclause 13.5, “<SQL procedure statement>”, GR2), specifies that a new statement execution context is created whenever an <SQL procedure statement> is executed. Note also that in Subclause 4.33.5, “SQL-statement atomicity and statement execution contexts”, we are told (last paragraph) that an SQL-transaction cannot be explicitly terminated within an atomic execution context. We conclude that the cited paragraph is relevant only when execution of a transaction-initiating statement (a) actually causes a transaction to be initiated, and (b) causes an exception to be raised of the special transaction rollback kind (this being the only way of implicitly terminating a transaction). In the light of this observation, we perceive the following problems: 1) There is no General Rule in, for example, Subclause 13.5, “<SQL procedure statement>”, to confirm the cited text.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					2) If execution of an SQL-statement causes an exception to be raised, then all changes to SQL-data and schemas are cancelled anyway. As the failing statement is also the one that initiated the transaction, the effect seems to be the same as that of a successful rollback, so what's the point in raising an additional exception expressing the fact that the transaction cannot be terminated? After all, the user executing the statement in question wasn't even trying to terminate the current transaction! We wonder if the rule was intended to cater for some eventuality other than the only one we can find. Even if the foregoing analysis proves to be refutable, it might be a good idea to add an explanation to Subclause 4.33.4, “SQL-statements and transaction states”. **Solution** None provided with comment.	
	NLD-P02-012		1-Major Technical	*P02-04.33.05, SQL-statement atomicity and statement execution contexts*	FND-924 The following Possible Problem has been noted: **Source:** WG3:HBA-029 **Possible Problem:** Subclause 4.33.5, “SQL-statement atomicity and statement execution contexts”, includes: The statement execution context brought into existence by the execution of an atomic SQL-statement or the evaluation of a <subquery> is an atomic execution context. The inclusion of "or the evaluation of a <subquery>", and the GRs of Subclause 7.15, “<subquery>”, that back it up, seem questionable. Isn't expression evaluation always atomic? The question also arises as to whether deletion of the questionable text (and GRs) would make any material difference to the standard. A search of the SQL:2003 Foundation FDIS for the word "atomic" reveals no GRs that are conditional upon the atomicity or non-atomicity of a statement execution context. Instead, there are some special GRs for <subquery> that enforce its atomicity by creating and destroying a savepoint level, and in Subclause 13.5, “<SQL procedure statement>”, for undoing any changes to SQLdata or schemas made execution of by an atomic statement that terminates with an exception. It seems, then, that the only effects caused by atomicity are to do with savepoints and database updates. But it appears that database updates are not possible during evaluation of a subquery, being outlawed by Subclause 7.13, “<query expression>”, SR23): 1) 23) A <query expression> QE shall not generally contain a <routine invocation> whose subject routine is an SQL-invoked routine that possible modifies SQL-data. Note that the BNF production for <subquery> is <left paren> <query	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					expression> <right paren> and a <query expression> cannot contain an SQL procedure statement. SR23) in combination with the GRs of Subclause 10.4, “<routine invocation>”, makes it impossible for an evaluation of an <routine invocation> caused by evaluation of a <subquery> to cause an SQL-data change statement to be executed. Therefore it is impossible for evaluation of a <subquery> to have any effect on SQL-data or schemas (possibly explaining the lack of a GR in Subclause 7.15, “<subquery>”, specifying that changes to SQL-data and schemas are to be cancelled). Therefore any savepoints established during evaluation of a <subquery> have to be ineffectual. Therefore there is no point in establishing a new, atomic, statement execution context for the evaluation of a <subquery>. But that's not all! Consider the <query expression> SELECT foo() FROM T, and suppose that there is some flaw in the reasoning that leads to the conclusion that the effect of the invocation of foo() cannot possibly depend on whether the current statement execution context is atomic. In that case SELECT foo() FROM T would not in general be equivalent to SELECT * FROM (SELECT foo() FROM T) T, for the shorter expression does not contain a <subquery>, whereas the longer one does. In general, the consequences of the effect of evaluation of a <query expression> possibly varying according to whether it is enclosed in parentheses are very unclear and would surely raise very awkward problems for optimisers. The foregoing analysis also brings into question the following sentence in Subclause 4.33.4, “SQLstatements and transaction states”: 1) If an <SQL-control statement> causes the evaluation of a <subquery> and there is no current SQLtransaction, then an SQL-transaction is initiated before evaluation of the <subquery>. Perhaps this is pointless, too. If it proves not to be, the wisdom of starting a transaction in the middle of executing an SQL-statement - - at an indeterminate point in that execution, to boot - - is surely questionable. In any case, we note that the sentence is not borne out by the GRs of Subclause 7.15, “<subquery>”. **Solution** Delete "or evaluation of a <subquery>" from the cited sentence of Subclause 4.33.5, “SQL-statement atomicity and statement execution contexts”; possibly delete the cited sentence of Subclause 4.33.4, “SQL-statements and transaction states”; delete GRs 1) ("Let OLDSEC ..."and 4) ("All savepoints ...") of Subclause 7.15, “<subquery>”. A search of the SQL:2003 Foundation FDIS for "<subquery>" reveals that a change might also be needed in Subclause 4.33.3, “SQL-statements and SQL-data access indication”. Other Parts of SQL:2003 have not been checked to see if they might be affected. But see WG3:HBA-041.	
	NLD-P02-013		1-Major	*P02-04.35.02,*	FND-972 The following Possible Problem has been noted:	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
			Technical	*Savepoints*	**Source:** WG3:SIA-031 = H2-2004-??? **Possible Problem:** Neither here nor anywhere else is there any definition of the term savepoint. Paragraph 5 of this Subclause contains: If a <rollback statement> references a savepoint SS, then all changes made to SQL-data or schema subsequent to the establishment of the savepoint are canceled, all savepoints established since SS was established are destroyed, and the SQL-transaction is restored to its state as it was immediately following the execution of the <savepoint statement>. The state of an SQL-transaction is not defined, nor is it referred to in Subclause 16.8, "<rollback statement>". Presumably what is being referred to is some (or all) of the SQL-session context. Which <savepoint statement> is not specified. Presumably the one that established *SS*. So perhaps "... following the establishment of *SS*"; or even "... as it was at that time". Perhaps what is really meant is something to the effect of, A savepoint is a preserved copy of (the values of specified elements) of the SQL-session context at the time a <savepoint statement> was executed, plus sufficient data to enable all subsequent changes to SQL-data or schemas in the current SQL-transaction to be canceled. When a <rollback statement> is executed, that contains a <savepoint specifier> *SS*, then all changes made to SQL-data or schema subsequent to the establishment of *SS* are canceled, all savepoints established since SS was established are destroyed, and elements of the SQL-session context are restored to the values that were preserved in *SS*. Paragraph 6 says: It is implementation-defined whether or not, or how, a <rollback statement> that references a <savepoint specifier> affects diagnostics area contents, the contents of SQL descriptor areas, and the status of prepared statements. This implementation-defined element is not mentioned in Annex B, "Implementation-defined elements" (or in Annex C, "Implementation-dependent elements"). **Solution:** Specify what happens in terms of the contents of the SQL-session context. **Solution** None provided with comment.	
	NLD-P02-014		1-Major Technical	*P02-04.37, SQL-sessions*	FND-954 The following Possible Problem has been noted: **Source:** WG3:ZSH-037R1/H2-2003-??? **Possible Problem:** WG3:FRA-045r4 proposed no changes to what is now WG3:ZSH-013, Subclause 4.37, "SQLsession". However, according to WG3:FRA-045r4, Section 2.1, "Authorization stack":	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					There is a stack of SQL-session contexts. There is one cell on this stack when the SQLsession begins. An additional SQL-session context is pushed on the stack for each <routine invocation>, and is removed when the <routine invocation> completes execution. There is no reference to this anywhere in this subclause, although there are various statements of the form "An SQL-session has a ...". Moreover, the list of SQL-session contents is incorrect and incomplete. The term "current SQL-session identifier" is listed, where the meaning of "current" is indicated in the following NOTE (55 in WG3:ZSH-013) and evidently used to distinguish the "current" SQL-session from dormant SQL-sessions. It is therefore probably intended to refer to the SQL-session identifier of the currently active (as opposed to dormant) SQL-session. If this surmise is correct, then the "current SQL-session user identifier" is missing. There is no reference to the authorization stack, though the two terms used to refer to the components of the only visible cell of that stack are mentioned. **Solution** None provided with comment.	
	NLD-P02-015		1-Major Technical	*P02-04.37.04, Execution contexts*	FND-955 The following Possible Problem has been noted: **Source:** WG3:ZSH-037R1/H2-2003-??? **Possible Problem:** This subclause contains the statement: There is always a statement execution context, a routine execution context, and zero or more trigger execution contexts. There is a significant and unnecessary inconsistency between the descriptions of routine execution contexts and trigger execution contexts. Consider what happens if an SQL-invoked routine R1 invokes another, R2. Are there now one or more than one routine execution contexts? The answer is clearly there is one in each of two levels of the stack of SQL-session contexts, as is made clear by Subclause 10.4, "<routine invocation>". Whether there is a routine execution context when no routine has been invoked is debatable: it could be (and indeed is) said that there is an empty one; or it could be said that there is none. In which case, it would be true to say that "there are zero or more routine execution contexts", as is said for trigger execution contexts. Consider now how it arises that there is more than one trigger execution context. The only case that springs to mind is that of the triggered action of a trigger T1, causing another trigger T2 to fire. In this case, each trigger will have a trigger execution context. However, it seems fairly clear that the triggered action of T2 cannot access the state changes in the trigger execution context of T1. Therefore, to say that there are, during the execution of T2, two trigger execution contexts, although true in a sense, is unhelpful. Moreover, we seem to be saying that these two trigger execution contexts are in	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					the same SQLsession context; unless, of course, T1 invokes a routine that causes T2 to fire, in which case a new SQL-session context is created, containing a new routine execution context. However, whether or not it contains, when created, the trigger execution context of T1, we are unable to discover. **Solution** None provided with comment.	
	NLD-P02-016		1-Major Technical	*P02-05.04, Names and identifiers*	FND-932 The following Possible Problem has been noted: **Source:** WG3:HBA-050R1 **Possible Problem:** SR19) of this subclause is: 19) An <identifier> that is a <correlation name> is associated with a table within a particular scope. The scope of a <correlation name> is either a <select statement: single row>, <subquery>, or <query specification> (see Subclause 7.6, "<table reference>"), or is a <trigger definition> (see Subclause 11.39, "<trigger definition>"). Scopes may be nested. In different scopes, the same <correlation name> may be associated with different tables or with the same table. The inclusion of <subquery> is puzzling. For consider that if such a scope is contained in a <subquery>, then it must also be wholly contained in some <query specification> contained in that <subquery>. Furthermore, a <subquery> that contains more than one <query specification> cannot possibly constitute the scope of any <correlation name>. For example: `( SELECT * FROM T1 UNION SELECT * FROM T2 )` The scope of any correlation name defined with such a <subquery> would be confined to the particular <query specification> in which it is defined. A scalar expression could be added to the <subquery> that includes an outer reference, but the <correlation name> used in that reference would have a wider scope than the <subquery>. The inclusion of <query specification> is also suspect, because the scope of a <correlation name> is not necessarily a whole <query specification>. **Solution** None provided with comment.	
	NLD-P02-017		1-Major Technical	*P02-05.04, Names and identifiers*	FND-946 The following Possible Problem has been noted: **Source:** WG3:HBA-042 = H2-2003-___ **Possible Problem:** This subclause says (regarding local temporary tables): Something needs to be said, in either or both of the Syntax Rules and General Rules of Subclause 5.4, "Names and identifiers", about how a <table name> identifies a created temporary table. As a minimum, a reference to a created local temporary table must be prohibited in any <schema routine> R, because, by the time R is invoked, it cannot be regarded as being syntactically contained in the <SQLclient module definition>	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					that contained the <externally-invoked procedure> which created it. **Solution** As a minimum, a Syntax Rule should be added, to the effect that: 1) If <table name> identifies a created local temporary table, then <table name> shall not be contained in a <schema routine>. Note: This appears to remove the need for any reference to created local temporary tables in Subclause 10.4, "<routine invocation>", General Rule 5) d) i), which is a problem for PSM.	
	NLD-P02-018		2-Minor Technical	*P02-06.01, <data type>*	FND-729 The following Language Opportunity has been noted: **Source:** WG3:YGJ-112 (SQL/MM YGJ-023), Paul Cotton for WG4, July 6, 1999, and Paul Scarponcini via email on 6 July 1999 **Language Opportunity:** According to YGJ-112: "REF types need to be scoped; i.e., the table(s) they refer to must be explicitly provided. If a column is of type REF type, the scope may be defined at table creation time. If the column is of type UDT which contains REF type attributes, then the scope must be declared when the UDT is created. The SQL/MM Part 3: Spatial standard defines the UDTs for spatial data. The standard is unable to predict in which tables the referenced information will be stored; this is a function of database design. Therefore, column scoping must be expanded to support deeply nested references, i.e., REF types within a UDT or ARRAY. This would allow a user, when creating tables, to define the scope of a UDTs REF type as part of the column definition for a column of type UDT." When a <reference type> is used as the data type of an attribute of a structured type, the <scope clause> must be specified when the encompassing user-defined type is defined. It is a Language Opportunity to be able to specify the <scope clause> of the "nested" <reference type>s when a column is defined on the encompassing user-defined type. Paul Scarponcini added: This applies to ARRAYs as well (e.g., an ARRY of REF, and ARRAY of UDTs having REF attributes. The resultant syntax may be quite messy, as different REFs within the column may have different scopes. Would it be worth considering reversing the scope specification: when the reference dtable is created, specify that it shall be included in the scope for a particular column, rahter than specifying the referenced table when the referencing column is specified? **Solution** None provided with comment.	
	NLD-P02-019		2-Minor Technical	*P02-06.01, <data type>*	FND-730 The following Language Opportunity has been noted: **Source:** WG3:YGJ-112 (SQL/MM YGJ-023) and Paul Cotton for WG4, July 6, 1999	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Language Opportunity: According to YGJ-112: "A second limitation of SQL 99 with respect to REF types is that they only achieve uni-directional "pointers"." A REF type value may be de-referenced to obtain the instance to which it refers. It is a Language Opportunity to provide direct support for determining all instances of a REF type which refer to a particular instance. **Solution** None provided with comment.	
	NLD-P02-020		2-Minor Technical	*P02-06.01, <data type>*	FND-812 The following Language Opportunity has been noted: **Source:** WG3:PER-098R1/H2-2001-059 **Language Opportunity:** Perhaps Feature S096, “Optional array bounds”, can be folded in Feature S091, “Basic array support”. **Solution** None provided with comment.	
	NLD-P02-021		2-Minor Technical	*P02-06.04, <value specification> and <target specification>*	FND-692 The following Language Opportunity has been noted: **Source:** DBL:CWB-081/X3H2-98-068 **Language Opportunity:** Although there is provision for refining a <value expression> of row type or structured type, there is no provision for refining a <target specification>. As a result, a field of a row or an attribute of a structured type cannot be passed as output or in/out argument of an SQL-invoked routine, or used in other target contexts. This problem is partially remedied in PSM <assignment statement>. Possibly the support for refined targets can be adapted from PSM and moved to Foundation. **Solution** None provided with comment.	
	NLD-P02-022		2-Minor Technical	*P02-06.04, <value specification> and <target specification>*	FND-723 The following Language Opportunity has been noted: **Source:** WG3:FRA-132/X3H2-98-694 **Language Opportunity:** Currently we have no capability to treat an <element reference> as a <target specification>. This precludes their use as output arguments of routine invocations, for example. The same observation can be made of <field reference>, <dereference operation>, <reference resolution>, and <method invocation> (some of these subject to the restriction that the method must be a mutator). (Lest you object that [Fred is] thinking of allowing surreptitious updates to column values by referencing them as output arguments of a routine invocation, be it noted that these expressions can also be used with parameters and variables.) However, [Fred believes] that the general solution to this problem is to introduce a notion of l-values and r-values, as in the specification of C.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution None provided with comment.	
	NLD-P02-023		1-Major Technical	*P02-06.06, <identifer chain>*	**WG3-P02-001** Consider the expression SELECT * FROM T T1 WHERE C1 = (SELECT MAX (C1) FROM T T2 WHERE T1.C2 > C1). It is surely indisputable that the two references to C1 in the subquery are syntactically legal and are references to T2.C1, according to the normal block-scoping rules that are commonly used in SQL implementations. And yet SR8) appears to make them illegal. SR8)a)ii) is applicable: ii) … [the <identifier chain>] shall be contained in the scope of one or more range variables whose associated tables include a column whose <column name> is equivalent to I_1 or in the scope of a <routine name> whose associated <SQL parameter declaration list> includes an SQL parameter whose <SQL parameter name> is equivalent to I_1. Let the phrase *possible scope tags* denote those range variables and <routine name>s. In the example, C1 is contained in the scope of both T1 and T2. The continuation of this subrule is a Case whose first subrule is: 1) If the number of possible scope tags in the innermost scope containing a possible scope tag is 1 (one), then let *IPST* be that possible scope tag. Now, if this condition were true in our example, and the single possible scope tag were T2, then all would be well, but unfortunately that does not appear to be the case. The innermost scope containing a possible scope tag for C1 consists of two fragments: SELECT MAX (C1) and WHERE T1.C2 > C1 (see Subclause 7.6, "<table reference>", SR5). How many of the two possible scope tags for C1 are "in" this scope? If "in" means "contained in", then the answer appears to be one, but the one in question is T1 (contained in the <where clause>), not T2. If on the other hand "in" means "that are in scope in", then the answer is two, for both T1 and T2 are in scope. Of course, "in" is not intended to mean either of those things; in fact, it is clear under this close examination that "in the innermost scope" is not an appropriate phrase here at all. Having shown that "in the innermost scope" is not appropriate, we now show that "containing a possible scope tag" isn't appropriate either. Consider the following slightly simpler example: SELECT * FROM T WHERE C1 = (SELECT MAX (C1) FROM T). How many possible scope tags do we have now? Well, MAX (C1) is in the scope of the T that is defined in the outer <from clause> and it is also in the scope of the other T that is defined in the <subquery>'s <from clause>. Do we have two possible scope tags that are both named T, or do we have just one possible scope tag with two distinct reasons for it being a possible scope tag? In any case, whether we have one or two, how many are "in the innermost scope containing a possible scope tag"? The scope of the T defined in the <subquery> is just SELECT MAX (C1), which contains no possible scope tags at all. The scope of the T defined in the	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					outer <from clause> consists of the fragments SELECT * and WHERE C1 = (SELECT MAX (C1) FROM T), which happens to contain T, though not the T that has this scope! It seems that when the same range variable name is used for two or more different purposes (and necessarily in that case with different scopes in each case), and when a column reference lies within each of those scopes, only the one *having* the innermost of those scopes is applicable (and so that one is applied). And when two or more *different* range variables are used, as in our first example, then they are all applicable but it is again the one *having* the innermost scope that is applied, provided, of course, that there is exactly one range variable qualifying as a possible scope tag, whose scope is the innermost of the scopes containing the column reference. **Solution** None provided with comment.	
	NLD-P02-024		2-Minor Technical	*P02-06.09, <set function specification>*	FND-819 The following Language Opportunity has been noted: **Source:** WG3:PER-044R1/H2-2000-619 **Language Opportunity:** The proponents of multiargument GROUPING function believe that it is a trivial extension of the single argument function, and therefore does not warrant a separate feature. This could be achieved by simply deleting the Conformance Rule that creates Feature T433, “Multiargument GROUPING function”, thereby allowing all GROUPING functions to fall under Feature T431, “Extended grouping capabilities”. **Solution** None provided with comment.	
	NLD-P02-025		2-Minor Technical	*P02-06.12, <cast specification>*	**WG3-P02-002** SR10) prohibits the containment of a <collate clause> in the target <data type>, *TD*. When a <data type> is specified, this is clear, but it is not so clear when a <domain name> is specified, in which case SR1) defines *TD* to be “the <data type> of the domain”. The BNF for <domain definition> (Subclause 11.24) doesn’t include a <data type>, though it does include a <predefined type>. A domain descriptor is said to include a data type descriptor, but note carefully that *every* character data type descriptor contains the fully qualified name of a collation. Note that a <predefined type> might include a <collate clause>, and also that if the <domain definition> contains a <collate clause>, then that is considered to be equivalent to the containment of that <collate clause> in the <predefined type>. Some tidying appears to be needed, but what the intended rule is in the case of casting to a domain needs to be determined before a precise redrafting can be proposed. Note that a related problem exists in the SQL:2007 WD for SQL/XML, Subclause 6.4, <XML cast specification>, SR10), so this P.P. might eventually need to be cloned as a CD ballot comment against Part 14.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution None provided with comment.	

	NLD-P02-026		2-Minor Technical	*P02-06.12, <set function specification>*	FND-693 The following Language Opportunity has been noted:
					Source: FCD1/1998 NLD-P02-017, DBL:CWB-132/X3H2-98-187
					Language Opportunity:
					We do not understand SR 4). If an outer reference is permitted at all, surely it should be permitted any number of times, just as literals and host variable names can occur any number of times. We would add that we see no reason to prohibit outer references altogether. For example, if SUM(OUTER.C1) is legal, surely SUM(OUTER.C1+OUTER.C1) is also legal. Besides, why should column references that are not outer references be prohibited as soon as there is an outer reference? SR 4) of Subclause 6.9, "<set function specification>", says:
					4) The <value expression> simply contained in <set function specification> shall not contain a <set function specification> or a <subquery>. If the <value expression> contains a column reference that is an outer reference, then that outer reference shall be the only column reference contained in the <value expression>.
					We agree that the above rule is overly restrictive. However, we believe this rule was adopted in SQL-92 to prohibit query formulations of the form:
					`SELECT *`
					`FROM t1`
					`GROUP BY ...`
					`HAVING ... ( SELECT c21`
					`FROM t2`
					`GROUP BY ...`
					`WHERE ... ( SELECT c3`
					`FROM t3`
					`WHERE SUM ( t1.c12 + t2.c22 ) > ...`
					`)`
					`)`
					In the above example, outer references from multiple levels are being referenced in the same aggregate function. Semantically, this does not make sense and must be prohibited.
					Solution
					None provided with comment.	

	NLD-P02-027		2-Minor Technical	*P02-06.15, <subtype treatment>*	FND-816 The following Language Opportunity has been noted: **Source:** WG3:PER-099/H2-2001-061 **Language Opportunity:** Perhaps Feature S162, "Subtype treatment for references", can be folded into Feature S161, "Subtype treatment". **Solution** None provided with comment.	
	NLD-P02-028		2-Minor	*P02-06.15,*	FND-829 The following Language Opportunity has been noted:	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
			Technical	*<subtype treatment>*	**Source:** WG3:PER-186/H2-2001-??? **Language Opportunity:** WG3:PER-099 extended <subtype treatment> so that an expression of type REF(*t1*) would be TREATed as one of type REF(*t2*) if *t2* is a subtype of *T1*. It was noted that, in that case, it should also be possible to TREAT: — An expression of type *t1* ARRAY[*n*] as one of type *t2* ARRAY[*n*]. — An expression of type *t1* MULTISET as one of type *t2* MULTISET. — An expression of type ROW(..., *f1 t1*, ...) as one of type ROW(..., *f1 t2*, ...). In the ROW case, it might even be possible to support TREATment over more than one field. For example, an expression of the type ROW(..., *f1 t1*, ..., *f2 t1*, ...) might be TREATable as ROW(..., *f1 t1*, ..., *f2 t2*, ...), as ROW(..., *f1 t2*, ..., *f2 t1*, ...), or as ROW(..., *f1 t2*, ..., *f2 t2*, ...), even though SQL does not (at the time of writing this Language Opportunity) support multiple inheritance in general. In the ROW case, it would also be necessary to decide whether field names must match as indicated in these examples. **Solution** None provided with comment.	
	NLD-P02-029		2-Minor Technical	*P02-06.28, <string value expression>*	FND-858 The following Language Opportunity has been noted: **Source:** WG3:ICN-054R2 = H2-2002-___ **Language Opportunity:** The term "character string operands" was used to replace a previously undefined term "components" in SR2. Is this the correct terminology to use? **Solution** None provided with comment.	
	NLD-P02-030		2-Minor Technical	*P02-06.34, <boolean value expression>*	FND-920 The following Language Opportunity has been noted: **Source:** WG3:ZSH-129 = H2-2002-___ **Language Opportunity:** The rules for known-not-null conditions in SR3) are more complicated than most implementations are prepared to implement, and not necessary for most users. The full implementation of known not null should be placed in a conformance feature. Without the feature, a much simpler definition should apply. **Solution** None provided with comment.	
	NLD-P02-031		2-Minor Technical	*P02-06.35, <array value expression>*	FND-808 The following Language Opportunity has been noted: **Source:** (was Possible Problem FND736) WG3:PER-171/H2-2001-??? (FCD1/2000 NLD-P02-027), from WG3:YGJ-074/X3H2-99-164R1 **Language Opportunity:** The ability to extract a subarray of an array would be useful. Such an ability would also satisfy a separate Language Opportunity to be able to truncate an array.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution None provided with comment.	
	NLD-P02-032		2-Minor Technical	*P02-07.04, <table expression>*	FND-756 The following Language Opportunity has been noted: **Source:** WG3:YGJ-069r1 = H2-99-155r3 and WG3:BHX-096/H2-2000-248R1 **Language Opportunity:** It might be useful to be able to filter windowed results based on the values of <OLAP function>, most likely through a new clause analogous to <where clause> and <having clause>, but following <window clause> **Solution** None provided with comment.	
	NLD-P02-033		2-Minor Technical	*P02-07.09, <group by clause>*	FND-610 The following Language Opportunity has been noted: **Source:** DBL:LGW-146/X3H2-97-349 **Language Opportunity:** Continuing work is needed to complete object support as outlined in "Providing Rich Query Functionality" (DBL:LHR-078 = X3H2-95-462) with regard to expanding GROUP BY to permit naming of grouping expressions and allowing those names to be used in the query. The ability to group the result of a table expression by the value of expressions is important to many applications. The ability to name these grouping expressions and use those names to retrieve the results of the grouping column cum expression in the select list of the table expression is equally important to avoid applications having to repeat the expression (giving opportunity for errors) in the select list. **Solution** None provided with comment.	
	NLD-P02-034		2-Minor Technical	*P02-07.12, <query specification>*	FND-528 The following Language Opportunity has been noted: **Source:** DBL:MAD-170/X3H2-96-544R1, point 2.1, FCD1/1998 CAN-P02-031, DBL:CWB-132/X3H2-98-187 **Language Opportunity:** DBL:MAD-170/X3H2-96-544R1, point 2.1, noted: The definition of a possibly nullable result column in the Syntax Rules of Subclause 7.12, "<query specification>", is broader than necessary, in that an aggregate of a column that is known not nullable is regarded as possibly nullable. For example, SUM(EMP.EMPNO) is defined as possibly nullable, even if EMP.EMPNO is declared NOT NULL. DBL:CWB-132/X3H2-98-187 added: The problem description makes the assumption that a <set function specification>, for example SUM(EMPNO), is known not nullable when EMPNO is known not nullable. However, GR 3)b) of Subclause 6.9, "<set function specification>", makes it clear that (with the exception of COUNT) <set function specification>s return NULL when they are applied to an empty table. Hence, we assume that <set function specification>s are possibly nullable,	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					except for COUNT. And, that is what SR 12) of Subclause 7.12, "<query specification>", specifies. Hence, we believe that there is no problem with SR 12) of Subclause 7.12, "<query specification>". **Solution** None provided with comment.	
	NLD-P02-035		2-Minor Technical	*P02-07.12, <query specification>*	FND-908 The following Language Opportunity has been noted: **Source:** P02, SQL/Foundation, Subclause 7.12, "<query specification>", CR 4) and CR 8) **Language Opportunity:** Conformance Rule 4) as formulated does not impose a restriction on the user writing SQL and as such does not follow the required model for Conformance Rules. Fred Zemke in an email to Stephen Cannan dated 2002-10-17 wrote: Subclause 7.12, "<query specification>" CR 4) - this is an example of the occasional practice of using the CRs to alter the definition of a defined term. This practice seems borderline to me. On the one hand, the CRs are regarded as merged with the SRs whenever the designated feature is absent, and definitions appear in the SRs, so it would seem possible to make a redefinition in a CR. On the other hand, does a redefinition constitute a limitation on the user? I think the better approach is the one taken regarding functional dependencies, for example, Subclause 7.12, "<query specification>", CR 3). This could have been done by defining a term such as 'group-invariant <value expression>' in the SRs, saying that all derived columns in the SELECT list of a grouped query must be group-invariant <value expression>s, and then the CR would alter the definition of groupinvariant <value expression>. Instead the approach taken is essentially to carve out two categories of derived column: the kind permitted in the SELECT list of a grouped query by the SRs, and the more restricted kind permitted by the CRs. Returning to 7.12 CR 4), the path would be to define two notions of updatable, using the more liberal one in the SRs and restricting to the more conservative one in the CRs. In fact, we already have two terms, updatable and simply updatable. The conclusion I am coming to is that this CR should be deleted, and, in any subclause that uses 'updatable' in an SR, there should be a CR that restricts to 'simply updatable' unless Feature T111 is present. Conformance Rule 8) as formulated does not impose a restriction on the user writing SQL and as such does not follow the required model for Conformance Rules. This rule should be deleted, and, in any subclause that uses 'updatable' in an SR, there should be a CR that restricts the use of UNION unless Feature T111 is present. **Solution** None provided with comment, but the body of the comment outlines a solution..	See Comment
	NLD-P02-036		2-Minor Technical	*P02-07.15, <subquery>*	FND-936 The following Language Opportunity has been noted: **Source:** WG3:HBA-050R1	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Language Opportunity: HBA-050 shows that the need for the BNF term <subquery> is nothing like as strong as it once might have been, and has given rise to a certain amount of difficulty and confusion. Perhaps it would be better to dispose of the term altogether (though <scalar subquery>, <row subquery>, and <table subquery> almost certainly need to be retained) and treat parenthesized <query expression>s in similar style to our treatment of parenthesized <value expression>s. Any proposal to address this Language Opportunity should of course check for existing uses of <subquery> in Parts other than Foundation. **Solution** None provided with comment.	
	NLD-P02-037		2-Minor Technical	*P02-08.02, <comparison predicate>*	FND-909 The following Language Opportunity has been noted: **Source:** WG3:ZSH-155 = H2-2002-___ **Language Opportunity:** The Syntax Rules convert all comparison predicates so that they only use < and =. The GRs for comparison of user-defined types spell out rules for > and other comparisons even though they have been transformed away. NOTE 167 following the GR claims that these unreachable GRs are there for informational purposes. In the case of RELATIVE order, there are some strong assumptions being made that *RF*(*X*,*Y*) = -*RF*(*Y*,*X*); otherwise, the system breaks down. We should document what are the expectations for the relative order function somewhere. We do not find such documentation either in <user-defined ordering function> or in Concepts. **Solution** None provided with comment.	
	NLD-P02-038		2-Minor Technical	*P02-09.03, Data types of results of aggregations*	FND-836 The following Language Opportunity has been noted: **Source:** WG3:YYJ-030R2 = H2-2001-___ and WG3:ZSH-155 = H2-2002-___ **Language Opportunity:** This subclause uses terms that are less precise than they should be. Specifically, the term result data type and data type of the result, without specifying the result of what. The first sentence of Function says: "Specify the result data type of the result of an aggregation ...". Moreover the term aggregation does not suggest the sense in which it is used here, having since been used extensively in the context of OLAP, see subclause 04.17.03 "Aggregate functions". A better title would be Data types of results of n-adic operations. Were this title adopted, the first sentence could be rewritten as, for example, Let IDTS be a set of data types specified in an application of this Subclause, and let O be the operation. **Solution** None provided with comment.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
	NLD-P02-039		2-Minor Technical	*P02-09.05, Type precedence list determination*	FND-709 The following Language Opportunity has been noted: **Source:** WG3:YGJ-021 **Language Opportunity:** Paper DBL:BBN-168 added a Syntax Rule to Subclause 11.50, "<SQL-invoked routine>", to prohibit the use of ROW because there is nothing in Subclause 9.5, "Type precedence list determination", to handle the type precedence requirements of anonymous row types. **Solution** None provided with comment.	
	NLD-P02-040		1-Major Technical	*P02-10.04, <routine invocation>*	FND-857 The following Possible Problem has been noted: **Source:** DCOR/2002, USA-STC-031 **Possible Problem:** There is no definition of how to pass booleans or LOBs to external programs. More generally, the question of how to convert any SQL type to a host language type at the interface to an SQL-invoked routine has never been addressed. Probably it was assumed that the same mechanism as was already defined for module language and embedded language applied, but in fact there are no rules to back up this assumption. If this assumption is correct, then the rules in Subclause 13.4, "Calls to an <externally-invoked procedure>", are probably appropriate. Perhaps they should be placed in a separate subclause so they can be referenced by both <routine invocation> and also <externally invoked procedure>. See also paper WG3:PER-176. **Solution** None provided with comment.	
	NLD-P02-041		1-Major Technical	*P02-10.04, <routine invocation>*	FND-956 The following Possible Problem has been noted: **Source:** WG3:ZSH-037R1/H2-2003-??? **Possible Problem:** 5) Preserve the current SQL-session context CSC and create a new SQL-session context RSC derived from CSC as follows: ... This appears to specify what happens to every element of an SQL-session context when a new SQL-session context is created. However, it does not say what happens to: — The zero or more trigger execution contexts — The values of all valid locators — The text defining the SQL-path (which in any case seems somewhat redundant, since the SQL-path is taken care of) — The SQL-session collations, if any — The text defining the default transform group name — The text defining the user-defined type name-transform group name pair for each userdefined type explicitly set by the user It would at least be clearer if it said:	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					5) Preserve the current SQL-session context CSC and create a new SQL-session context RSC as follows: ... **Solution** None provided with comment.	
	NLD-P02-042		2-Minor Technical	*P02-11, Schema definition and manipulation*	FND-710 The following Language Opportunity has been noted: **Source:** WG3:YGJ-021 **Language Opportunity:** A RENAME TABLE statement has been strongly desired for a very long time and any users will be expecting to see it in SQL3. **Solution** None provided with comment.	
	NLD-P02-043		2-Minor Technical	*P02-11, Schema dewfinition and manipulation*	FND-694 The following Language Opportunity has been noted: **Source:** DBL:CWB-114/X3H2-98-169 **Language Opportunity:** The current choices for <drop behavior>, RESTRICT and CASCADE, are too limiting. CASCADE is so sweeping that the user must hesitate to use it, not knowing what may be dropped. RESTRICT, on the other hand, is so limited that the user must find all dependencies and drop them in the proper order. There is a third model, based on the notion of invalidation. With this model, a dependent definition does not block a drop; instead, the dependent object is simply marked invalid. Later usage of an invalid object causes its recompilation, which may very well succeed since the cause of invalidation may have been repaired. **Solution** None provided with comment.	
	NLD-P02-044		2-Minor Technical	*P02-11.03, <table definition>*	FND-822 The following Language Opportunity has been noted: **Source:** WG3:PER-104/H2-2001-085R1 **Language Opportunity:** The ability to specify options for inheriting column default and identity column properties, as in the <like clause>, would also be beneficial for the <as subquery clause>. **Solution** None provided with comment.	
	NLD-P02-045		2-Minor Technical	*P02-11.03, <table definition>*	FND-874 The following Language Opportunity has been noted: **Source:** WG3:DRS-095 **Language Opportunity:** Since in section 1.1.2 [of WG3:DRS-095] we gave reasons for determining the <reference generation> implicitly, it would be most convenient if the <column constraint definition>s necessary for derived reference representations were implicit, and determined by examination of the corresponding user-defined type descriptor.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution None provided with comment.	
	NLD-P02-046		2-Minor Technical	*P02-11.05, <default clause>*	FND-642 The following Language Opportunity has been noted: **Source:** DBL:LGW-152/X3H2-97-352 (also DBL:LGW-023/X3H2-97-044, SEQ# 222, CAN-F-062, converted to LO by WG3:BHX-038/H2-2000-018R3) **Language Opportunity:** It might be useful to allow default values for row types, perhaps by using row constructors. **Solution** None provided with comment.	
	NLD-P02-047		2-Minor Technical	*P02-11.05, <default clause>*	FND-712 The following Language Opportunity has been noted: **Source:** WG3:YGJ-021 and WG3:PER-098R1/H2-2001-059 **Language Opportunity:** It is not possible to specify default values for columns or attributes of an array type, a multiset type, a reference type, a row type, or a user-defined type. **Solution** None provided with comment.	

	NLD-P02-048		2-Minor Technical	*P02-11.08, <referential constraint definition>*	FND-349 The following Language Opportunity has been noted:
					Source: WG3:YGJ-074/X3H2-99-164R1 (Bill Kelley noted the following Language Opportunity, which has been modified by Fred Zemke)
					Language Opportunity:
					For collections types, referential integrity is not definable for elements of collections.
					Example: Assume table EMPLOYEE has PRIMARY KEY EMP_ID of type INTEGER:
					`CREATE TABLE MANAGER (`
					`EMPNO INTEGER,`
					`MANAGES INTEGER ARRAY[20] )`
					Here "MANAGES" refers to a set of employees, but there is no way to say that they should reference employees. That is, if one were to write:
					`CREATE TABLE MANAGER (`
					`EMPNO INTEGER,`
					`MANAGES INTEGER ARRAY[20] REFERENCES EMPLOYEE )`
					then EMPLOYEE.EMPNO must be a column of array type, and teh constraint says that the array value in MANAGER.MANAGES must either be null or be equal to an array value in EMPLOYEE.EMPNO.
					What is needed is a new syntax, perhaps:
					`CREATE TABLE MANAGER (`
					`EMPNO INTEGER,`
					`MANAGES INTEGER ARRAY[20] ELEMENT REFERENCES EMPLOYEE(EMPNO) )`
					ELEMENT REFERENCES would mean that each array element of MANAGER.MANAGES must either be null or equal value in	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					EMPLOYEE.EMPNO. **** Editor's Note (number 15) **** (Editor's note: In my opinion, Bill is simply trying to solve the problem using the wrong tools. INTEGER ARRAY[n] is meant to have elements of integers, not elements of employee IDs...which is a different thing altogether.) **Solution** None provided with comment.	
	NLD-P02-049		2-Minor Technical	*P02-11.10, <alter table statement>*	FND-747 The following Language Opportunity has been noted: **Source:** WG3:RTM-028/X3H2-99-252R1 **Language Opportunity:** It might be useful to have an option so that a conventional (SQL-92) table can evolve to become a table of type. However, any such proposal must avoid the pitfalls noted during development of SQL:1999 for evolution to a table of "named row type" (to use the terminology current before structured types were introduced). The proposal must account for the <reference type specification> of the user-defined type. If <reference generation> is DERIVED, it may be necessary to require a unique constraint or primary key constriant on the appropriate columns. If <references generation> is USER GENERATED, it may be necessary to require that the table has no rows. Probably the self-referencing column must be added to the table as part of its evolution to a table of structured type. It is unlikely that the unaltered table will have as its first column a reference to the very type to which the table will be evolving. And, if perchance that condition were met, what would be do with the previously existing values in that column? **Solution** None provided with comment.	

	NLD-P02-050		1-Major Technical	*P02-11.22, <view definition>*	FND-933 The following Possible Problem has been noted:
					Source: WG3:HBA-050R1
					CR4) of this subclause is:
					4) Without Feature F751, “View CHECK enhancements”, conforming SQL language shall not contain <view definition> that contains a <subquery> and contains CHECK OPTION.
					This is suspect. Even if it really was intended to rule out, as it apparently does, examples of the following form (note the <subquery>):
					`CREATE VIEW V AS`
					`SELECT ...`
					`FROM ( SELECT ... FROM ... ) AS T`
					`WHERE ...`
					`WITH CHECK OPTION`
					then surely it should also be ruling out examples of the following equivalent form:
					`CREATE VIEW V AS`	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By

					`WITH A AS ( SELECT ... FROM ... )`
					`SELECT ...`
					`FROM A`
					`WHERE ...`
					`WITH CHECK OPTION`
					But it doesn't. (Note the lack of any <subquery>.)
					A minor additional point is that "WITH CHECK OPTION" would be safer than just "CHECK OPTION", in case WITHOUT CHECK OPTION is ever added to the language.
					Solution
					None provided with comment.	

	NLD-P02-051		1-Major Technical	*P02-11.30, <drop domain statement>*	FND-938 The following Possible Problem has been noted: **Source:** WG3:HBA-028 **Possible Problem:** GR1)c) refers to "the explicit or implicit <constraint name list>". The BNF production for <drop domain definition> does not included a <constraint name list>, nor do the Syntax Rules specify an implicit one in any circumstances. **Solution** Delete GR1)c) and edit the lead-in of GR1)d) as shown here: d) For every domain constraint descriptor included in the domain descriptor of D [begin deletion] whose <constraint name> is not contained in the excluded constraint list[end deletion]:	
	NLD-P02-052		2-Minor Technical	*P02-11.39, <trigger definition>*	FND-611 The following Language Opportunity has been noted: **Source:** DBL:LGW-146/X3H2-97-349 **Language Opportunity:** SQL3 should consider adding syntax to allow the user to specify the ordering in which triggers on the same effect should be fired. **Solution** None provided with comment.	

	NLD-P02-053		2-Minor Technical	*P02-11.41, <user-defined type definition>*	FND-603 The following Language Opportunity has been noted:
					Source: DBL:LGW-131/X3H2-97-293, 24 July, 1997; also USA-081 in first CD ballot for SQL/Foundation
					and WG3:YGJ-074/X3H2-99-164R1
					Language Opportunity:
					Subclause 11.41, "<user-defined type definition>", contains a Syntax Rule reading:
					6)g) [A user-defined type] shall not be based on itself.
					This syntax rule prevents the UDT facility from modeling a recursively-defined data type such as "Tree".
					Here is a simple example of a UDT definition that is not possible because of that SR:
					`CREATE TYPE Tree (`
					`node_value INTEGER,`	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By

					`left_subtree Tree,`
					`right_subtree Tree )`
					Solution
					None provided with comment.	

	NLD-P02-054		2-Minor Technical	*P02-11.50, <SQL-invoked routine>*	FND-713 The following Language Opportunity has been noted: **Source:** WG3:YGJ-021 **Language Opportunity:** Currently all parameters must be of some specific concrete type. There needs to be a mechanism to declare that a parameter is a character string of arbitrary, unspecified type, at least when invoking PSM. (And there should be some mechanism within PSM to interrogate the character set and length of a character string parameter). Otherwise the subject routine rules allow you to resolve to the same PSM routine no matter what the parameter's character set, but when the function is invoked, you will get an error when trying to assign the input argument to the parameter's type if the input argument's character set is different from the one declared in the function's signature. There should also be a mechanism to declare that the return type of a function is determined by a parameter's type. **Solution** None provided with comment.	
			1-Major Technical	*P02-12.01, <grant statement>*	**WG3-P02-003** GRs 4)b), 4)c), 4)d), 4)e), 5), 6), though curiously not 7) all contain the phrase "[f]ollowing the successful execution of the <grant statement> …". Given that GRs are to be evaluated in the order in which they are written, and that "successful execution" usually (though not always) means that the last GR has been reached, that wording doesn't seem to make sense. Note that the BNF for <grant statement> specifies that it is either a <grant privilege statement> or a <grant role statement>. The rule evaluation order specified in Framework, Subclause 6.3.3.3, makes it clear that the rules for the contained statements are applied before the rules for the containing statement. Perhaps, then, the wording we have questioned should be changed to something to the effect of "following the successful execution of the contained statement"; but if the contained statement fails, then doesn't evaluation of the rules of the containing statement end too? In that case, each of these phrases can simply be deleted. Note also that GRs 1), 2), 3), 4) 5), 6) and 7) all specify the execution of "the following <grant statement> …". Whoever drafts a solution to this problem might like to check that there is no infinite recursion going on here. It might be that "the following <grant statement>" should better be "the following <grant privilege statement>" or "the following <grant role statement>", as applicable. **Solution** None provided with comment.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
	NLD-P02-055		2-Minor Technical	*P02-12.07, <revoke statement>*	FND-734 The following Language Opportunity has been noted: **Source:** Email from Fred Zemke, 1999-06-09 and WG3:ZSH-155 = H2-2002-___ **Language Opportunity:** The OLAP Amendment has created a new kind of dependency, of a view, *etc.*, containing an OLAP function that references a user-defined ordering in its ORDER BY clause, which is dependent on the userdefined ordering. <drop routine statement> has been edited to account for this dependency; does any other statement need to be edited? **Solution** None provided with comment.	
	NLD-P02-056		2-Minor Technical	*P02-12.07, <revoke statement>*	FND-911 The following Language Opportunity has been noted: **Source:** WG3:ZSH-155 = H2-2002-___ **Language Opportunity:** Syntax Rule 36) says: 36) If RESTRICT is specified, then there shall be no abandoned privilege descriptor, abandoned view, abandoned table constraint, abandoned assertion, abandoned domain constraint, lost domain, lost column, lost schema, and no descriptor that includes an impacted data type descriptor, impacted collation, impacted charater set, abandoned user-defined type, forsaken column decriptor, forsaken domain descriptor, or abandoned routine descriptor. This SR has several problems: — It is unclear whether there should be a comma following "schema", though we recognize that a schems is a descriptor. (Note: This problem has been fixed by the addition of "and no" between "schema," and "descriptor".) — It is unclear whether the object of "includes" is a nested list. (Note: This problem has been resolved by making it clear that it is a nested list.) — The terms used to refer to impacted, etc., objects are inconsistent with those used to so designate them. While it is **descriptors** that are said to be abandoned, impacted, etc., this rule referes to "impacted **columns**", etc. — Several possible candidates for inclusion in the list are absent for no obvious reason; they include abandoned table descriptor, abandoned trigger descriptor, and contaiminated column descriptor. We suggest improving the clarity by using a possibly nested bullet list. **Solution** None provided with comment.	
	NLD-P02-057		1-Major Technical	*P02-12.07, <revoke statement>*	FND-979 The following Possible Problem has been noted: **Source:** WG3:SIA-018 = H2-2003-429 **Possible Problem:** WG3:SIA-018 introduced the notions of view components, view component privilege descriptors, and view privilege dependency descriptors pertaining to a	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					given view. WG3:SIA-018 proposed explicit rules that specify the creation of view privilege dependency descriptors, but failed to specify explicit rules that specify the destruction of view privilege dependency descriptors. Although a view privilege dependency descriptor can be assumed to be destroyed whenever either its supporting privilege descriptor or the dependent privilege descriptor is destroyed, the standard would be clearer if this were done in the appropriate place(s) in the GRs of Subclause 12.7, "<revoke statement>". **Solution** None provided with comment.	
	NLD-P02-058		2-Minor Technical	*P02-13.01, <SQL-client module definition>*	FND-921 The following Language Opportunity has been noted: **Source:** FCD1/2002, GBR-P02-485 **Language Opportunity:** None of the GRs in this Subclause relate to the creation of an SQL module. Moreover, General Rule 4) relates to the invocation of an externally-invoked procedure. **Solution** None provided with comment.	
	NLD-P02-059		2-Minor Technical	*P02-13.03, <externally-invoked procedure>*	FND-844 The following Language Opportunity has been noted: **Source:** WG3:YYJ-034 = H2-2001-___ **Language Opportunity:** The use of savepoint levels, introduced by WG3:PER-061 and extended by WG3:YYJ-034, still does not cover the case of externally-invoked procedures. **Solution** None provided with comment.	
	NLD-P02-060		2-Minor Technical	*P02-13.05, <SQL procedure statement>*	FND-925 The following Possible Problem has been noted: **Source:** WG3:HBA-029 **Possible Problem:** Subclause 13.5, "<SQL procedure statement>", includes two GRs to the effect that if the statement being executed is an atomic one, then all changes to SQL-data and schemas are cancelled. Shouldn't this be conditional on whether the current execution context is atomic, rather than on the statement type? Not that this would make any material difference, but as things stand there appears to be no point in the final sentence of GR2) of this Subclause: 1) 2) A statement execution context NEWSEC is established for the execution of S. Let OLDSEC be the most recent statement execution context. NEWSEC becomes the most recent statement execution context. NEWSEC is an atomic execution context, and therefore the most recent atomic execution context, if and only if S is an atomic SQL-statement. Although there are GRs in various subclauses that do enforce atomicity where it is required, none of these rules references the atomicity or non-atomicity of an execution context.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution Either delete all references to atomicity of execution contexts, or change GRs 5)a)ii) and 5)b)ii)1) of Subclause 13.5, "<SQL procedure statement>", to be conditional on the atomicity of the current statement execution context. Probably the former solution is to be preferred, in view of the specific GRs in several places that refer to savepoint levels and undoing changes to SQL-data and schemas. But see WG3:HBA-041.	
	NLD-P02-061		2-Minor Technical	*P02-13.06, Data type correspondences*	FND-815 The following Language Opportunity has been noted: **Source:** WG3:PER-107/H2-2001-115 **Language Opportunity:** Table 18, "Data type correspondences for COBOL", maintains that the COBOL type corresponding to BOOLEAN is PICTURE X. Before the deletion of the BIT type (by paper WG3:PER-107/H2-2001-115), Subclause 20.5, "<embedded SQL COBOL program>", maintained that the declaration "PIC X USAGE IS BIT" could be used either to correspond to a bit string or to a BOOLEAN. This was flawed because the embedded COBOL processor needs to know what SQL type to assign to an embedded variable declaration. After the deletion of the BIT type, there appears to be no support for BOOLEAN in Subclause 20.5, "<embedded SQL COBOL program>", not even in a buggy Syntax Rule. Note that it will not do to overload "PICTURE X" as either CHAR(1) or BOOLEAN, for the same reason that it was not acceptable to overload "PIC X USAGE IS BIT" as either BIT(1) or BOOLEAN. Perhaps "USAGE IS BOOLEAN" is in order. **Solution** None provided with comment.	
	NLD-P02-062		1-Major Technical	*P02-14.07, <delete statement: searched>*	FND-939 The following Possible Problem has been noted: **Source:** WG3:HBA-028 **Possible Problem:** GR9) is as follows: 9) Each <subquery> in the <search condition> is effectively executed for each row of T and the results are used in the application of the <search condition> to the given row of T. If any executed <subquery> contains an outer reference to a column of T, then the reference is to the value of that column in the given row of T. NOTE 496 — 368 - - "outer reference" is defined in Subclause 6.7, "<column reference>". As GR5) already says that the <search condition> is "applied to [sic -- evaluated for might be better] each row of T", perhaps GR9) isn't needed at all. If its existence is justified by the apparently inadequate definition of outer reference in the referenced Subclause 6.7, then surely it would be better to fix SR4) of that Subclause to cater for outer references that are not contained in <table	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					expression>s. The first sentence of GR9) refers to each <subquery> in the <search condition>. It is questionable whether what it says is really applicable to every <subquery> contained in the <search condition>, regardless of how deeply nested it is. In any case, the sentence is imprecise and inappropriately worded. We don't execute subqueries, and the meaning of "results are used in the application of" is unclear. Similar problems exist in the following rules: • Subclause 7.8, "<where clause>", GR3) • Subclause 7.10, "<having clause>", GR2) • Subclause 14.9, "<merge statement>", GR6)a)i)1) • Subclause 14.11, "<update statement: searched>", GR5)a)ii) and GR5)b)ii) **Solution** None provided with comment.	
	NLD-P02-063		2-Minor Technical	*P02-14.08, <insert statement>*	FND-715 The following Language Opportunity has been noted: **Source:** WG3:YGJ-021 **Language Opportunity:** When a row of a table that has a system-generated column is inserted, the application has no way to access the newly generated value. This was not an issue when only explicit values were inserted by the application. **Solution** None provided with comment.	
	NLD-P02-064		2-Minor Technical	*P02-14.10, <update statement: positioned>*	FND-717 The following Language Opportunity has been noted: **Source:** WG3:YGJ-021 **Language Opportunity:** The Format for <update target> does not provide a way to set a field of an anonymous row type. Seemingly the only way to update column of an anonymous row type is to replace the entire column, which will be awkward in many instances. For example, suppose I only want to update the STREET portion of an ADDRESS column. Looks like I have to use UPDATE T SET ADDRESS = ROW (:STREETVAR, T.CITY, T.STATE, T.ZIP); This means the query writer has to repeat the entire definition of the anonymous row in the query, which can be quite laborious, as well as hiding the simplicity of what the user is actually doing. Also, we must support all kinds of nesting of anonymous rows and UDTs. **Solution** None provided with comment.	
	NLD-P02-065		2-Minor Technical	*P02-14.10, <update statement: positioned>*	FND-724 The following Language Opportunity has been noted: **Source:** WG3:FRA-093/X3H2-98-628) **Language Opportunity:** The <simple value specification> immediately contained in an <update target> of a <set clause> specifying the array element of the target column to be updated	See comment

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					should be a <value specification> rather than a <simple value specification>. This would allow the use of a <dynamic parameter specification> which is currently prohibited because a <simple value specification> cannot be a <dynamic parameter specification>. General Rules 14)a)ii)5)c) of <update statement: positioned> and <update statement: searched> will cause an exception to be raised if a null value is passed as a <value specification> so no change is necessary to preclude a null value. **Solution** — Changes to Subclause 14.10, "<update statement: positioned>": • Revise the BNF for <update target>, replacing <simple value specification> with <value specification>. • Replace <simple value specification> with <value specification> in Syntax Rule 10), General Rule 14) and Conformance Rule 2). — Changes to Subclause 14.11, "<update statement: searched>": • Replace <simple value specification> with <value specification> in Syntax Rule 9) and General Rule 14).	
	NLD-P02-066		2-Minor Technical	*P02-14.10, <update statement: positioned>*	FND-809 The following Language Opportunity has been noted: **Source:** (was Possible Problem FND-737) WG3:PER-171/H2-2001-???, FCD1/2000 NLD-P02-063 (from WG3:YGJ-074/X3H2-99-164R1) **Language Opportunity:** There is no ability to truncate an array. Assigning NULL to the last element of an array does not decrease the length of the array. **Solution** None provided with comment.	
	NLD-P02-067		2-Minor Technical	*P02-14.12, <set clause list>*	FND-922 The following Language Opportunity has been noted: **Source:** WG3:ZSH-163 = H2-2003-___ **Language Opportunity:** Impossible to Update Different Parts of the Same Column SR 7) prohibits the same column name from appearing more than once in the list of SET clauses. This means that the user who wishes to use the shorthands available for assigning to fields of rows is rather severely restricted, unacceptably so, in our opinion. The problem does not arise in connection with assignment to attributes of UDT values, thanks to the ingenious SR 6). **Solution** None provided with comment.	
	NLD-P02-068		2-Minor Technical	*P02-16.02, <set transaction statement>*	FND-912 The following Language Opportunity has been noted: **Source:** WG3:ZSH-155 = H2-2002-___ **Language Opportunity:** The standard does not specify a maximum for <number of conditions>. Presumably there is an implementation-defined or -dependent maximum value of <number of conditions>. For example, we could add the following GR after	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					GR 2): 2) If <number of conditions> exceeds an implementation-dependent maximum number of conditions, then an exception condition is raised: *invalid condition number*. We must also add an entry in either the implementation-defined or the implementation-dependent Annex. Note: WG3:ICN-001 recorded "After some discussion, the consensus was that the condition should be a warning and that a good solution to the comment should involve adding an extra field to the diagnostics area, giving the current transaction's maximum number of conditions." **Solution** None provided with comment.	
	NLD-P02-069		1-Major Technical	*P02-16.04, <set constraints mode statement>*	FND-919 The following Possible Problem has been noted: **Source:** WG3:ZSH-031R3 = H2-2002-___ **Possible Problem:** The subclause is silent with regard to the checking of constraints when the constraints mode is set to IMMEDIATE. Turning to Subclause 16.7, "<commit statement>", we see that there is an expectation that SET CONSTRAINTS ALL IMMEDIATE has the effect of checking all constraints and that this effect takes place between GR5) and GR6) of that subclause (as opposed to any vague notion of "at the end of the statement"). The implications for referential constraints that specify referential actions are not clear, especially in the case of referential actions that are triggering events. **Solution** None provided with comment.	
	NLD-P02-070		1-Major Technical	*P02-16.04, <set constraints mode statement>*	FND-940 The following Possible Problem has been noted: **Source:** WG3:HBA-028 **Possible Problem:** If a <set constraints mode statement> is used to change the current mode of some constraint from deferred to immediate, it might happen that the database fails to satisfy that constraint. In this case, an exception is raised, but the database remains unchanged, so every subsequent statement will fail with the same exception, apart from one that sets the relevant constraint's mode back to DEFERRED or one that makes some change to the database to return it to a consistent state. (One such statement is COMMIT, which turns itself into ROLLBACK if constraints are not satisfied.) **Solution** None provided with comment.	
	NLD-P02-071		4-Minor Editorial	*P02-16.05, <savepoint statement>*	FND-973 The following Possible Problem has been noted: **Source:** WG3:SIA-031 = H2-2004-??? **Possible Problem:** General Rule 4) of this Subclause is:	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					1) A savepoint is established in the current savepoint level and at the current point in the current SQLtransaction. S is assigned as the identifier of that savepoint. This is not sufficiently specific. It doesn't even say that sufficient data is preserved for the successful execution of a subsequent <rollback statement>. **Solution** Specify what happens in terms of the contents of the SQL-session context.	
	NLD-P02-072		1-Major Technical	*P02-16.07, <commit statement>*	FND-941 The following Possible Problem has been noted: **Source:** WG3:HBA-028 **Possible Problem:** Just before submitting this paper we discovered, in Subclause 16.7, "<commit statement>", the following GR: 6) Case: a) If any constraint is not satisfied, then any changes to SQL-data or schemas that were made by the current SQL-transaction are canceled and an exception condition is raised: *transaction rollback — integrity constraint violation*. b) If the execution of any <triggered SQL statement> is unsuccessful, then any changes to SQL-data or schemas that were made by the current SQL-transaction are canceled and an exception condition is raised: *transaction rollback — triggered action exception*. c) If any other error preventing commitment of the SQLtransaction has occurred, then any changes to SQL-data or schemas that were made by the current SQL-transaction are canceled and an exception condition is raised: *transaction rollback* with an implementation-defined subclass value. d) Otherwise, any changes to SQL-data or schemas that were made by the current SQL-transaction are eligible to be perceived by all concurrent and subsequent SQL-transactions. This seems problematical. Case (a) is possibly okay, catering for any deferred constraints, though there is an opportunity to make it more precise using text similar to what HBA-028 proposed for constraint checking in Subclause 13.5. Regarding case (b), it is not clear how a <commit statement> can possibly cause a <triggered SQL statement> to be invoked. Regarding case c), it is not clear what "other error preventing commitment" refers to. Perhaps an informative note is needed. **Solution** None provided with comment.	
	NLD-P02-073		1-Major Technical	*P02-16.07, <commit statement>*	FND-970 The following Possible Problem has been noted: **Source:** WG3:SIA-023 = H2-2004-??? **Possible Problem:** After acceptance of WG3:SIA-023, Subclause 16.7, "<commit statement>", GR 9)a) is: a) If <commit statement> contains AND CHAIN, then an SQL-transaction is	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					initiated. Any branch transactions of the SQL-transaction are initiated with the same access mode, isolation level, and diagnostics area limit as the corresponding branch of the SQL-transaction just terminated. Before acceptance of WG3:SIA-023, it was: a) If AND CHAIN was specified, then a new SQL-transaction is initiated with the same access mode, isolation level, and diagnostics area limit as the SQL-transaction just terminated. Any branch transactions of the SQL-transaction are initiated with the same access mode, isolation level, and diagnostics area limit as the corresponding branch of the SQL-transaction just terminated. The simplification of the first sentence was made possible by WG3:SIA-023's simpler approach to the setting of transaction characteristics. The problems lie in the second sentence. It is not clear what "[a]ny branch transactions of the SQLtransaction" refers to. It seems as if it refers to things that exist, and yet "initiated" suggests that they are to be brought into existence. Also, no mention is made of the initial constraint modes of branch transactions. Maybe the intent is to specify that in every SQL-session containing a branch transaction of the transaction just terminated, a branch transaction is initiated. But it is not clear how branch transactions come into existence in the first place. Subclause 4.35, "SQL-transactions", mentions the possibility of their existence without explaining how they arise. A second point that might need to be considered by anybody attempting to address this P.P. concerns the initial constraint modes for the new transaction initiated by AND CHAIN. GR5) clearly specifies that all constraint modes are immediate, but given the inadequacy we have noted in GR9) it might be that this was not really intended. It seems more intuitive to have constraint modes reinitialised to their declared initial states, as when AND CHAIN is not specified. If that was really the intent (and perhaps what has actually been implemented by implementations supporting Feature F721, "Deferred constraints", then we might have to consider accepting an incompatible change. **Solution** None provided with comment.	
	NLD-P02-074		1-Major Technical	*P02-16.07, <commit statement>*	FND-976 The following Possible Problem has been noted: **Source:** WG3:SIA-030 = H2-2004-??? **Possible Problem:** Subclause 16.7, "<commit statement>", SR 3) is: 1) For every open cursor that is not a holdable cursor *CR* in any SQL-client module associated with the current SQL-transaction, the following statement is implicitly executed: `CLOSE CR` Exactly which cursors are to be closed under this rule is not clear, even when we safely assume that it does not mean cursors other than holdable ones in any SQL-client module associated with the current SQL-transaction. We surmise	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					that “in any SQL-client module associated with the current SQL-transaction” is redundant. We suspect that it is intended to refer to every (non-holdable) cursor open in the current SQL-session, but as it stands it appears to exclude any global extended dynamic cursor allocated during execution of the body of an SQL-invoked routine. Although Subclause 4.22, “SQL-client modules”, does say that an SQL-client module includes “Zero or more cursors”, there is no statement, there or elsewhere in Clause 4, “Concepts”, to the effect that every cursor is in some sense “in” some SQL-client module. Subclause 4.35, “SQL-transactions”, does contain the statement that “Each SQL-client module that executes an SQL-statement of an SQL-transaction is associated with that SQL-transaction”, but it's not clear what it means for an SQL-client module to be associated (or not) with a particular SQL-transaction, nor is it clear what it means to say “Each SQL-client module that executes an SQL-statement ...”, since statements executed in SQL-invoked procedures might or might not be included. The General Rules of Subclause 16.8, “<rollback statement>”, suffer from similar problems, and we additionally note a curious difference between GR2)e) (“All open cursors … are closed”) and its counterpart for the ROLLBACK TO SAVEPOINT case, GR3)g), which explicitly specifies execution of certain <close statement>s. Possible Problem **FND-975** describes another problem with the cited General Rule. It might be desirable to address both problems in a single change proposal. **Solution** None provided with comment.	

	NLD-P02-075		1-Major Technical	*P02-18.01, <set session characteristics statement>*	FND-971 The following Possible Problem has been noted:
					Source: WG3:SIA-023 = H2-2004-???
					Possible Problem:
					The BNF production for <set session characteristics statement> is:
					`<set session characteristics statement> ::=`
					`SET SESSION CHARACTERISTICS AS <session characteristic list>`
					`<session characteristic list> ::=`
					`<session characteristic> [ { <comma> <session characteristic> }... ]`
					`<session characteristic> ::= <transaction characteristics>`
					According to this BNF, the following are both legal <set session characteristics statement>s:
					`SET SESSION CHARACTERISTICS AS TRANSACTION`
					`READ ONLY,`
					`ISOLATION LEVEL SERIALIZABLE,`
					`DIAGNOSTICS SIZE 2`
					`SET SESSION CHARACTERISTICS AS`	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					TRANSACTION READ ONLY, TRANSACTION ISOLATION LEVEL SERIALIZABLE, TRANSACTION DIAGNOSTICS SIZE 2 If this strange-looking syntax was not actually intended, then it should be corrected. **Solution** <session transaction characteristics> shall contain at most one <isolation level>, at most one <transaction access mode>, and at most one <diagnostics size>. <set session characteristics statement> ::= SET SESSION CHARACTERISTICS AS <session transaction characteristics> <session transaction characteristics> ::= TRANSACTION <transaction mode> [<comma> <transaction mode> ...] Replace the Syntax Rules by: 1) <session transaction characteristics> shall contain at most one <isolation level>, at most one <transaction access mode>, and at most one <diagnostics size>. Replace the General Rules by: 1) Let *STC* be the <session transaction characteristics>. Let *ESC* be the enduring session characteristics of the current SQL-session. 2) If *STC* contains an <isolation level> *IL*, then the isolation level of *ESC* is set to the <level of isolation> contained in *IL*. 3) If *STC* contains an <access mode> *AM*, then the access mode of *ESC* is set read-only or read-write according to whether *AM* contains READ ONLY or READ WRITE, respectively. 4) If *STC* contains a <diagnostics size> *DS*, then the condition area limit of *ESC* is set to the <number of conditions> contained in *DS*.	
	NLD-P02-076		1-Major Technical	*P02-18.02, <set session user identifier statement>*	FND-977 The following Possible Problem has been noted: **Source:** WG3:SIA-026R3 = H2-2004-??? **Possible Problem:** In SQL/Foundation, GR5) is: 5) If the current user identifier and the current role name are restricted from setting the user identifier to *V*, then an exception condition is raised: *invalid authorization specification*. It is not clear how to interpret GR5) in the case where current user and current role do not both exist. Furthermore, suppose they do both exist and just one of them is restricted from setting "the user identifier" to *V*. The rule is written to require both of them to be so restricted for that exception condition to be raised. This seems a little arbitrary and we wonder if that was what was really intended. **Solution** None provided with comment.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
	NLD-P02-077		2-Minor Technical	*P02-19, Dynamic SQL*	FND-726 The following Language Opportunity has been noted: **Source:** WG3:FRA-126R1 and WG3:PER-098R1/H2-2001-059 **Language Opportunity:** There is no way to retrieve a locator for an array, a multiset, or a UDT without having pre-knowledge of the type of data to be accessed because the rules for <get descriptor statement> require that the data type of the <simple target specification> "match" that represented by the item descriptor area when retrieving DATA. For UDT locators, "match" implies that the UDT for which the locator was declared be the same as that specified in the SQL item descriptor area. For array locators and multiset locators, "match" implies that the element data types be the same. The only way to declare a host variable appropriately is to know in advance what UDTs, arrays, or multisets will be accessed. This is unacceptable for dynamic SQL. A similar problem exists with reference types. **Solution** None provided with comment.	
	NLD-P02-078		2-Minor Technical	*P02-19.06, <prepare statement>*	FND-926 The following Possible Problem has been noted: **Source:** WG3:HBA-040 **Possible Problem:** General Rule 10) is: 1) 10) If <statement name> is specified for the <SQL statement name>, P is not a <cursor specification>, and <statement name> is associated with a cursor C through a <dynamic declare cursor>, then an exception condition is raised: *dynamic SQL error — prepared statement not a cursor specification*. This rule is redundant: all it does is warn the user that he won't be able to open the dynamic cursor; unless, of course, he subsequently executes a <prepare statement> with the same <statement name> and an <SQL statement variable> whose value is a <cursor specification>. The check belongs on <dynamic open statement>. But see WG3:HBA-041. **Solution** None provided with comment.	
	NLD-P02-079		1-Major Technical	*P02-19.11, <output using clause>*	FND-949 The following Possible Problem has been noted: **Source:** WG3:HBA-048 = H2-2003-___ **Possible Problem:** General Rule 3) of this Subclause is: Case: a) If PS is a <dynamic select statement> or a <dynamic single row select statement>, then the <output using clause> describes the <target specification>s for the <dynamic fetch statement> or the <execute statement>. Let *D* be the degree of the table specified by *PS*. The use here of the BNF non-terminal <target specification> is inappropriate in the case that <into descriptor> is specified.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution None provided with comment.	
	NLD-P02-080		1-Major Technical	*P02-19.11, <output using clause>*	FND-950 The following Possible Problem has been noted: **Source:** WG3:HBA-048 = H2-2003-___ **Possible Problem:** General Rule 6)c) of this Subclause is: If the <output using clause> is used in a <dynamic fetch statement>, then let LTDT be the data type on the most recently executed <dynamic fetch statement>, if any, for the cursor CR. It is implementation-defined whether or not an exception condition is raised: *dynamic SQL error — restricted data typ attribute violation* if any of the following are true: It is not clear to what "the data type on the most recently executed <dynamic fetch statement>, if any, for the cursor CR" is intended to refer to. There is no data type on (or even in or of) a <dynamic fetch statement>. We suspect that what is meant is: It is implementation-defined whether or not you're allowed to fetch into a locator on one fetch from CR, but not on the next, or vice versa. Moreover, Annex B, "Implementation-defined elements" contains no entry for this Subclause. **Solution** None provided with comment.	
	NLD-P02-081		1-Major Technical	*P02-19.11, <output using clause>*	FND-951 The following Possible Problem has been noted: **Source:** WG3:HBA-048 = H2-2003-___ **Possible Problem:** General Rule 6) of this Subclause contains two subrules that cause a locator to be generated. The structure is: 6) For 1 (one) δ i δ D: Case: i) If TDT [Target Data Type] is a locator type, then: 1) If SV is not the null value, then a locator L that uniquely identifies SV is generated and is the value TV of the i-th <target specification>. Case: If <into descriptor> is specified, then ... Case: Otherwise, [TVT is assumed to be a locator of some sort] Case: If TV is not the null value, then: Case: If TYPE indicates a locator type, then a locator L that uniquely identifies TV is generated and the value of DATA is set to an implementation-dependent four-octet value that represents L. This appears to be generating a locator of a locator. And why "... an	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					implementation-dependent four-octet value that represents ..."? **Solution** None provided with comment.	
	NLD-P02-082		2-Minor Technical	*P02-19.11, <output using clause>, and 19.10, <input using clause>*	FND-952 The following Language Opportunity has been noted: **Source:** WG3:HBA-048 = H2-2003-___ **Language Opportunity:** Subclause 19.10, "<input using clause>", Syntax Rule 1) is: 1) The <general value specification> immediately contained in <using argument> shall be either a <host parameter specification> or an <embedded variable specification>. and Subclause 19.11, "<output using clause>", Syntax Rule 1) is: 1) The <target specification> immediately contained in <into argument> shall be either a <host parameter specification> or an <embedded variable specification>. It is thus not currently possible for an SQL parameter to be either a <using argument> or an <into argument>. **Solution** None provided with comment.	
	NLD-P02-083		1-Major Technical	*P02-19.17, <dynamic fetch statement>*	FND-948 The following Possible Problem has been noted: **Source:** WG3:HBA-048 = H2-2003-___ **Possible Problem:** General Rule 2) of this Subclause is: 2) The General Rules of Subclause 19.11, "<output using clause>", are applied to the <dynamic fetch statement> and the current row of *CR* as the retrieved row. Subclause 19.11, "<output using clause>", doesn't mention either "the current row" or "the retrieved row". **Solution** The invocations of the General Rules of this Subclause should be regularised.	
	NLD-P02-084		1-Major Technical	*P02-19.22, <preparable dynamic delete statement: positioned> and P02-19.23, <preparable dynamic update statement: positioned>*	FND-930 The following Possible Problem has been noted: **Source:** WG3:HBA-040 **Possible Problem:** Both subclauses contain <scope option> in the Format, yet say nothing about it in either Syntax Rules or General Rules. Both subclauses contain a Syntax Rule: 2) All Syntax Rules of Subclause 14.n, "<xx statement: positioned>", apply to the <preparable dynamic xx statement: positioned>, replacing "<declare cursor>" with "<dynamic declare cursor> or <allocate cursor statement>" and "<xx statement: positioned>" with "<preparable dynamic xx statement: positioned>". Neither <xx statement: positioned> refers to a <declare cursor> (they did once), and if they did, it is difficult to understand how <allocate cursor statement>	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					could be relevant, since the cursor it creates cannot be referenced by a <cursor name>. **Solution** None provided with comment.	
	NLD-P02-085		2-Minor Technical	*P02-20.01, <embedded SQL host program>*	FND-364 The following Language Opportunity has been noted: **Source:** WG3:YGJ-074/X3H2-99-164R1 **Language Opportunity:** There is a problem for precompilers when the issue of overlapping and non-disjoint scopes for host variables, etc. comes into play. In addition, there are problems caused by things like C macros and the C #ifdef conditional facilities. **Solution** None provided with comment.	
	NLD-P02-086		1-Major Technical	*P02-20.01, <embedded SQL host program>*	FND-770 The following Possible Problem has been noted: **Source:** WG3:BHX-166 **Possible Problem:** Since multiple SQL data types map onto the same C data type in Table 17, “Data type correspondences for C”, in Subclause 13.6, “Data type correspondences”, SR22) of Subclause 20.1, “<embedded SQL host program>”, cannot correctly identify the corresponding SQL data type of a given C data type. The problem identified is caused by Table 17, “Data type correspondences for C”, in Subclause 13.6, “Data type correspondences”, that defines the mapping of C data types onto SQL data types. The table maps more than one SQL data type onto the same C data type. Hence, when the mapping table is used in reverse, a single C data types maps onto more than one SQL data type. Now, in case of syntax rule 22) of Subclause 20.1, “<embedded SQL host program>”, the SQL data type has to be determined while an <embedded SQL host program> is processed. Thus, the SQL data types can only be derived syntactically from the C data types based on Table 17, “Data type correspondences for C”, in Subclause 13.6, “Data type correspondences”. The solution of the problems would require a change of Table 17, “Data type correspondences for C”, in Subclause 13.6, “Data type correspondences”, such that a single SQL data type maps onto a single C data type. There might be an alternative solution which accesses the definition of a routine to find out the SQL data types rather than using the mentioned table. Both solutions result in major changes of the document and might also lead to compatibility issue. Hence, a real solution of the identified problems cannot be developed in the given timeframe. **Solution** None provided with comment.	
	NLD-P02-087		2-Minor Technical	*P02-20.05, <embedded SQL*	FND-947 The following Language Opportunity has been noted: **Source:** LO arising from WG3:HBA-038 = H2-2003-294	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
				COBOL program>	**Language Opportunity:** With the publication of COBOL 2002, there are opportunities for exploiting the new features in COBOL 2002 in specifying the data type correspondences for COBOL. Mapping SQL user-defined types to object capabilities in COBOL 2002 should also be investigated. **Solution** None provided with comment.	
	NLD-P02-088		2-Minor Technical	*P02-22.01, <get diagnostics statement>*	**WG3-P02-004** GR6)b) appears to assume that a <get diagnostics statement> specifies a single assignment, whereas in general it can specify several, these being possibly of both statement information items and condition information items. Some kind of "for each …" construct is needed in the phrasing of this rule. Arguably such treatment should really be applied to GRs 2) onwards, so that they become subrules of a single outermost rule, but it might be considered acceptable to let them stand and just fix GR6). **Solution** None provided with comment.	
	NLD-P02-089		1-Major Technical	*P02-24, Conformance*	Feature F121 Basic diagnostics management (or at least sufficient to return the information inherent in F491) should be included in Core SQL. **Solution** None submitted with comment	
	NLD-P02-090		1-Major Technical	*P02-24, Conformance*	Feature F391 Long Identifiers should be included in Core SQL. **Solution** None submitted with comment	
	NLD-P02-091		1-Major Technical	*P02-24, Conformance*	Feature F491 Constraint management should be included in Core SQL. **Solution** None submitted with comment	
	NLD-P02-092		1-Major Technical	*P02-24, Conformance*	Feature T051 Row types should be included in Core SQL. **Solution** None submitted with comment	
	NLD-P02-093		1-Major Technical	*P02-24, Conformance*	Feature T141 SIMILAR predicate should be included in Core SQL. **Solution** None submitted with comment	
	NLD-P02-094		1-Major Technical	*P02-24, Conformance*	Feature T351 Bracketed SQL comments should be included in Core SQL. **Solution** None submitted with comment	
	NLD-P02-095		1-Major Technical	*P02-24, Conformance*	UNICODE as a mandatory character set should be included in Core SQL. **Solution** None submitted with comment	
	NLD-P02-096		1-Major Technical	*P02-F, SQL feature taxonomy*	FND-935 The following Possible Problem has been noted: **Source:** WG3:HBA-050R1 **Possible Problem:**	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					In Table 35, “Feature taxonomy and definition for mandatory features”, row 134, the Description for Feature F131, “Grouped operations” is: — A grouped view is a view whose <query expression> contains a <group by clause> This contradicts the definition of grouped view that existed vacuously in SQL:1999 and has since been deleted. Furthermore, it doesn't seem to be an accurate summary of what Feature F131, “Grouped operations” really is. See FIPS 127-2, feature 13, for the proper definition. **Solution** None provided with comment.	
	NLD-P02-097		1-Major Technical	*P02-No particular location*	FND-772 The following Possible Problem has been noted: **Source:** WG3:BHX-118 **Possible Problem:** The proposal accepted in WG3:BHX-118 creates a new problem. It makes is possible for an externallyinvoked procedure invoked directly from the SQL-client to define a WITH RETURN cursor that is left open when the externally-invoked procedure returns to the SQL-client. This is at best meaningless, since the SQL-client has no way to do anything with that cursor, and at worst causes a problem with resource "leaks" related to unclosed cursors. **Solution** None provided with comment.	
	NLD-P02-098		2-Minor Technical	*P02-No particular location*	FND-918 The following Possible Problem has been noted: **Source:** WG3:ZSH-034R1 = H2-2002-___ **Possible Problem:** What does CURRENT_ROLE tell us? During execution of an SQL routine *R* whose security characteristic is DEFINER, an invocation of CURRENT_ROLE will return the authorization identifier (i.e., the role name) of the owner of *R*. If it were considered that a user might be interested in knowing what role was actually set by the most recent <set role statement>, then we would need a SESSION_ROLE, analogous to SESSION_USER. **Solution** None provided with comment.	
	NLD-P02-099		2-Minor Technical	*P02-No specific location*	FND-014 The following Language Opportunity has been noted: **Language Opportunity:** It was noted in conjunction with CAN-106 discussions that if one inserts a row in a view V1 but do not have INSERT privilege on the underlying view V2 that has a WITH CHECK OPTION constraint, then a *constraint violation* exception is raised; however, one can then not discover anything about that constraint! **Solution**	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					None provided with comment.	
	NLD-P02-100		2-Minor Technical	*P02-No specific location*	FND-055 The following Language Opportunity has been noted: **Language Opportunity:** It has been noted that schema manipulation requires no privileges, but depends directly on ownership of the schema. **Solution** None provided with comment.	
	NLD-P02-101		2-Minor Technical	*P02-No specific location*	FND-129 The following Language Opportunity has been noted: **Language Opportunity:** [Note from SLC] We use the terms "destroyed", "deallocated", "deleted", "released", and perhaps others in various places. Are these terms used consistently and could the number of such terms be reduced? **Solution** None provided with comment.	
	NLD-P02-102		2-Minor Technical	*P02-No specific location*	FND-134 The following Language Opportunity has been noted: **Language Opportunity:** [Note from SLC] The functions LOWER and UPPER might be better defined in terms of translations and collations so that they properly account for all character sets instead of only <simple Latin character>s. **Solution** None provided with comment.	
	NLD-P02-103		2-Minor Technical	*P02-No specific location*	FND-190 The following Language Opportunity has been noted: **Source:** Jim Melton **Language Opportunity:** Jim Melton said, in his response to TC LB X3H2-90-267: We believe that many implementations will have schema objects other than those specified in SQL2 (e.g., indexes, stored <module>s, etc.) that may depend on schema objects defined in SQL2. The DROP semantics for such implementations will depend on those implementation-defined objects as well as those specified in SQL2, yet the SQL2 DROP rules do not appear to make allowances for additional restrictions on DROP statements. The wording in SQL2 must be enhanced to allow for such additional restrictions. Paper X3H2-90-373 addressed this, but failed. X3H2 suggested that a broader proposal that addresses the general concept of implementation-defined objects that might restrict CASCADE operations would be acceptable. **Solution** None provided with comment.	
	NLD-P02-104		2-Minor Technical	*P02-No specific location*	FND-212 The following Language Opportunity has been noted: **Source:** LON-034/X3H2-90-333.1 **Language Opportunity:** The ISO SQL2 Editing Meeting in London noted that with the advent of a	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					default character set for domains and columns in a schema, there is an opportunity to change that default character set for the schema. This might, for example, involve an ALTER SCHEMA CHANGE CHARACTER SET statement. **Solution** None provided with comment.	
	NLD-P02-105		2-Minor Technical	*P02-No specific location*	FND-217 The following Language Opportunity has been noted: **Source:** Stephen Cannan **Language Opportunity:** Steve Cannan has noted: It might be necessary to redefine the actions of triggers so that certain actions survive an *unsuccessful* execution of an SQL statement. For example, a BEFORE DELETE trigger might be used to record *attempts* to alter a table for security reasons. It would therefore be necessary that the triggered action survive an error in the original statement. **Solution** None provided with comment.	
	NLD-P02-106		2-Minor Technical	*P02-No specific location*	FND-241 The following Language Opportunity has been noted: **Language Opportunity:** [From London] The following Opportunity exists: When counting the number of rows "affected" by an <SQL statement>, one might consider counting the rows that are affected by triggered statements, too (e.g., triggers and referential constraints). **Solution** None provided with comment.	
	NLD-P02-107		2-Minor Technical	*P02-No specific location*	FND-242 The following Language Opportunity has been noted: **Language Opportunity:** [From London] The following Opportunity exists: For language consistency, a correlation name should be permitted for the modified table in positioned and searched update and delete statements. **Solution** None provided with comment.	
	NLD-P02-108		2-Minor Technical	*P02-No specific location*	FND-268 The following Language Opportunity has been noted: **Language Opportunity:** During consideration of YOK-023/X3H2-92-252, following language opportunity was identified: The set of <identifier>s available as <regular character set identifier>s in the <similar predicate> (see Subclause 8.6, “<similar predicate>”) could profitably be enhanced to support additional character attributes (*e.g., ideographs, syllables, etc.*, as a result of internationalization work subh as that going on in SC22/WG20.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution None provided with comment.	
	NLD-P02-109		2-Minor Technical	*P02-No specific location*	FND-309 The following Language Opportunity has been noted: **Source:** Phil Shaw **Language Opportunity:** Local declarations of dynamic cursor names would seem like a straightforward extension to X3H2-93-056/YOK-034rev. **Solution** None provided with comment.	
	NLD-P02-110		2-Minor Technical	*P02-No specific location*	FND-317 The following Language Opportunity has been noted: **Source:** X3H2-93-445/MUN-160 **Language Opportunity:** The representation of SQL-paths in the Information Schema needs to be specified. **Solution** None provided with comment.	
	NLD-P02-111		2-Minor Technical	*P02-No specific location*	FND-327 The following Language Opportunity has been noted: **Source:** X3H2-93-370R1/MUN-170 **Language Opportunity:** Object-oriented applications that model the behavior of real-=world entities need the ability to add an existing object to a type or to remove it from a type without destroying the object. Existing persons become employees and later stop being employees while continuaing to exist as persons. This can be achieved with a modest extension of current facilities. The paper went on to add that a simple extension would be allow a constructor such as STUDENT() to accept an optional parameter whose value is an existing object that is to be made an instance of STUDENT (but only if it is in the type hierarchy with STUDENTs). **Solution** None provided with comment.	
	NLD-P02-112		2-Minor Technical	*P02-No specific location*	FND-426 The following Language Opportunity has been noted: **Source:** Paper X3H2-94-528/DBL:RIO-081 noted the following Possible Problem; WG3:BBN-155/X3H2-98-378 changed it to a Language Opportunity: **Language Opportunity:** This possibility (factoring out parts of <column definition>, <field definition>, ...) was pointed out as an opportunity in SOU-076, and we considered attempting it. However, although there seemed to be no problem with the BNF, we were unsure how to specify a default character set. Consider Syntax Rule 6) of <column definition>, which reads: 6) If a <data type> is specified, then:	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					a) Let *DT* be the <data type>. b) If DT is CHARACTER, CHARACTER VARYING, or CHARACTER LARGE OBJECT and does not specify a <character set specification>, then the <character set specification> specified or implicit in the <schema character set specification> of the <schema definition> that created the schema identified by the <schema name> immediately contained in the <table name> of the containing <table definition> or <alter table statement> is implicit. c) If *DT* is a <character string type> that identifies a character set that specifies a <collate clause> and the <column definition> does not contain a <collate clause>, then the <collate clause> of the <character string type> is implicit in the <column definition>. Now, apart from the fact that this masterpiece of prolicity probably has more angle brackets than it should have, it just doesn't seem to work anyway for a LOCAL DECLARED TABLE (which has MODULE instead of a <schema name>). Furthermore, the Syntax Rules for <SQL variable declaration> (in RIO-006, SQL/PSM) contain nothing corresponding to this rule. If it's needed here, is it not also needed there? We seem to need something rather more generic, such as "the character set of the relevant schema". The difficulty is specifying what we mean by "relevant" so as to cover all cases, but it should surely be possible. **Solution** None provided with comment.	
	NLD-P02-113		2-Minor Technical	*P02-No specific location*	FND-440 The following Language Opportunity has been noted: **Source:** Paul Cotton noted the following Language Opportunity in Ottawa, July, 1995 **Language Opportunity:** DBL:YOW-027 changed Subclause 13.4, “Calls to an <externally-invoked procedure>”, to define BOOLEAN parameters as zero (0) for FALSE and one (1) for TRUE for the C language. However, Subclause 6.12, “<cast specification>”, does not currently permit BOOLEAN source values to be cast to a target value of type exact numeric. This would appear to be inconsistent with the abovereferenced change. An opportunity exists to permit this cast. **Solution** None provided with comment.	
	NLD-P02-114		2-Minor Technical	*P02-No specific location*	FND-452 The following Language Opportunity has been noted: **Source:** DBL:YOW-102/X3H2-95-244 discussion **Language Opportunity:** The specification of the isolation levels is less precise and rigorous than it should be; as a result, the intent is somtimes misperceived and the details are often imsinterpreted.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution None provided with comment.	
	NLD-P02-115		2-Minor Technical	*P02-No specific location*	FND-453 The following Language Opportunity has been noted: **Source:** Steve Cannan noted the following Language Opportunity during discussion of DBL:YOW-055/X3H2-95-140: **Language Opportunity:** Rules such as Subclause 11.10, "<alter table statement>", Syntax Rule 2) ("The schema identified by...shall include the descriptor of *T*") would be unnecessary if the phrase "identified by" was defined to require existence. **Solution** None provided with comment.	
	NLD-P02-116		2-Minor Technical	*P02-No specific location*	FND-468 The following Language Opportunity has been noted: **Source:** X3H2-94-103/DBL:SOU-076 **Language Opportunity:** X3H2-94-103/DBL:SOU-076 only introduced a ROW_TYPE for SQL (i.e., for SQL variables, parameters, results, and columns). The host language data types are still the scalar types specified in SQL-86, SQL-89, and SQL-92. Thus, the proposal doesn't add the new SQL ROW_TYPE to the host language mappings for module language, embedded syntax, or external routine parameters. Support for host language ROW_TYPEs would require specifying the forms of host language record declarations that are recognized in embedded syntax, and adding such host language record types to the data type correspondences for embedded syntax, module language, and external routines. Such a proposal would presumably include the ability to reference such host language variables as targets of FETCH, SELECT, and assignment statements, as sources of INSERT, UPDATE, and assignment statements, and as arguments of IN, OUT, and INOUT parameters. See also Language Opportunities PSM-078 , and CLI-003 , BIND-003 . **Solution** None provided with comment.	
	NLD-P02-117		2-Minor Technical	*P02-No specific location*	FND-469 The following Language Opportunity has been noted: **Source:** X3H2-94-103/DBL:SOU-076 **Language Opportunity:** SQL3 table definitions include a new LIKE clause that lets you "copy" column definitions from existing tables: `CREATE TABLE EMP_DEPT (LIKE EMP, LIKE DEPT, OTHER_COLUMN CHAR(5))` A similar clause would seem useful for ROW_TYPE declarations. The clause would, however, need to be generalized somewhat to allow for specifying row expressions other than tables. **Solution** None provided with comment.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By

	NLD-P02-118		2-Minor Technical	*P02-No specific location*	FND-470 The following Language Opportunity has been noted:
					Source: X3H2-94-103/DBL:SOU-076
					Language Opportunity:
					As noted in PP FND-469 , the LIKE clause provides a shorthand for creating tables of similar formats.
					As described in X3H2-94-103/DBL:SOU-076, this proposal includes the ability to specify a ROW_TYPE as a DOMAIN or a DISTINCT TYPE (this results from definint ROW_TYPE as a <data type>). A possible follow-on proposal could extent CREATE TABLE to allow reference to ROW_TYPE domains and/or types:
					`CREATE DOMAIN NAME AS ROW_TYPE (FIRST CHAR(10), LAST CHAR(10));`
					`CREATE TABLE OF NAME;`
					There are several detailed questions that such a proposal would need to address. FOr example, can domain names and LIKE both be used in a CREATE TABLE? Can a DISTINCT TYPE be used in a CREATE TABLE?
					Solution
					None provided with comment.	

	NLD-P02-119		2-Minor Technical	*P02-No specific location*	FND-471 The following Language Opportunity has been noted: **Source:** X3H2-94-103/DBL:SOU-076 **Language Opportunity:** Given two rows, R1 and R2, a "concatenation" or "join" operator could be defined. For discussion, assume that it would be written with the operator ||. Then, if R1 has F1 fields and R2 has F2 fields, R1 || R2 would yield a row with F1+F2 fields, where the values of the first F1 fields are the values of the fields of R1 and the values of the last F2 fields are the values of the fields of R2. **Solution** None provided with comment.	
	NLD-P02-120		2-Minor Technical	*P02-No specific location*	FND-473 The following Language Opportunity has been noted: **Source:** X3H2-94-103/DBL:SOU-076 **Language Opportunity:** According to this paper, two ROW_TYPEs are equivalent (and assignable) if both have the same number of fields and every pair of fields in the same position have compatible types. A possible follow-on could consider an option for assignment and type equivalence rules based on the names (instead of the positions) of the fields, similar to the <corresponding specification> of <query expression>s. **Solution** None provided with comment.	
	NLD-P02-121		2-Minor Technical	*P02-No specific location*	FND-474 The following Language Opportunity has been noted: **Source:** X3H2-94-103/DBL:SOU-076	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Language Opportunity: A possible follow-on paper could extend the definition of ROW_TYPEs to allow constraints and default values. **Solution** None provided with comment.	
	NLD-P02-122		2-Minor Technical	*P02-No specific location*	FND-475 The following Language Opportunity has been noted: **Source:** X3H2-94-103/DBL:SOU-076 **Language Opportunity:** A possible follow-on paper could integrate the rules for ROW_TYPE comparisons in predicates into one single Subclause. **Solution** None provided with comment.	
	NLD-P02-123		2-Minor Technical	*P02-No specific location*	FND-519 The following Language Opportunity has been noted: **Source:** X3H2-96-111/DBL:MCI-098 **Language Opportunity:** The TRIGGERED_COLUMNS base table in the Definition Schema misses an opportunity to capture both the explicit UPDATE columns of a trigger and other explicit or implicit "referenced" columns of the trigger. Consider replacing the "TRIGGERED_COLUMNS base table" in the current specification with the following new base table and view: TRIGGER_COLUMN_USAGE base table This table would consist of 8 columns (instead of the 7 columns in the existing TRIGGERED_COLUMNS base table). 3 columns to identify the Catalog, Schema, and Name of a Trigger. 4 columns to identify the Catalog, Schema, Table, and Name of a Column. 1 column to indicate whether the named column is an explicit UPDATE column (specified in the <trigger column list> of an UPDATE <trigger event> of this trigger), an explicit "Contained" column (contained in the <triggered action> of this trigger), or an "Implicit" column (implicitly referenced because it happens to be a column in the subject table of an UPDATE Trigger specified without an explicit <trigger column list>). This 8-th column could also be used later to identify other kinds of column usage that may be the basis of a <trigger event>, e.g. SELECT (if triggers are extended to SELECT actions), or the actual column (or columns) that get updated by an INSTEAD OF trigger. TRIGGER_COLUMN_USAGE view This view would consist of the same 8 columns as in the base table, but would return only columns owned by the CURRENT_USER that are "referenced" in some trigger (either owned by the CURRENT_USER or by some other user). The 8-th column would tell the owner what kind of "reference" (i.e. UPDATE, Contained, or Implicit) is being made to his column by the identified trigger. The TRIGGER_COLUMN_USAGE view would make it possible for a given user to return a list of columns (owned by that CURRENT_USER) that are the	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					UPDATE Trigger columns of a trigger (possibly owned by some other user) defined in this catalog. This information is not derivable from the existing TRIGGERED_COLUMNS view because that view only returns triggers owned by the CURRENT_USER. The TRIGGERED_COLUMNS view (redefined over the new TRIGGER_COLUMN_USAGE base table) and the new TRIGGER_COLUMN_USAGE view could be used separately to answer all of a users legitimate Trigger questions. The TRIGGERED_COLUMNS view would return the UPDATE columns of triggers owned by the CURRENT_USER and the TRIGGER_COLUMN_USAGE view would return all catalog triggers that explicitly or implicitly "reference" a column owned by the CURRENT_USER. The first view would return the names of columns owned by other people that the given user had UPDATE privileges on, but never the names of triggers owned by other people, and the second view would return the names of triggers owned by other people but never the names of columns owned by other people. Both views are valuable to the user and contain information that a user has legitimate reason to know. **Solution** None provided with comment.	
	NLD-P02-124		2-Minor Technical	*P02-No specific location*	FND-521 The following Language Opportunity has been noted: **Source:** DBL:MCI-098/X3H2-96-111 **Language Opportunity:** The trigger descriptor defined in GR 2 of Subclause 11.39, "<trigger definition>", maintains an explicit collection of all column names referenced by the <triggered action> of the <trigger definition>. This makes the trigger descriptor different in style from a table constraint descriptor (see Subclause 11.6, "<table constraint definition>", GR2) or a view descriptor (see Subclause 11.22, "<view definition>", GR1), which only maintain this information implicitly. A table check consraint maintains the entire <search condition> of the Check and a view descriptor maintains the entire <query expression> that determines the view. It may be desirable to treat constraint, view, and trigger descriptors in a more homogeneous fashion. Alternatively, a trigger descriptor may just maintain the <triggered action> as part of the descriptor, rather than the "triggered action column set". If this is done instead, then Syntax Rule 5 and General Rule 1 of Subclause 11.18, "<drop column definition>", would have to be re-written to accommodate <triggered action> instead of "triggered action column set". **Solution** None provided with comment.	
	NLD-P02-125		2-Minor Technical	*P02-No specific location*	FND-587 The following Language Opportunity has been noted: **Source:** Hugh Darwen, 27 January, 1997 **Language Opportunity:**	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Currently, all <routine invocation>s that return values are deemed to be able to return a null. Hence, such results are automatically tagged as "possibly null". Wouldn't it be nice if you could say, when you define a function, "NEVER RETURNS NULL" or words to that effect? Then its invocations would have the nice "not nullable" characteristic. **Solution** None provided with comment.	
	NLD-P02-126		2-Minor Technical	*P02-No specific location*	FND-593 The following Language Opportunity has been noted: **Source:** DBL:LGW-063/X3H2-97-077, point 46. **Language Opportunity:** There are no provisions in SQL3 for packaging ADT families. This type of packaging is needed to support the creation of a package of ADTs and associated subtypes and routines. It would be useful to define access control at the package level rather than the individual ADTs or routines. It would also be useful to be able to isolate the package so that subject routine resolut8ion of routines inside the package can be restricted to only other routines within the package. This packaging could be accomplished with schemas or SQL-server modules, but neither mechanism is complete at this point. **Solution** None provided with comment.	
	NLD-P02-127		2-Minor Technical	*P02-No specific location*	FND-613 The following Language Opportunity has been noted: **Source:** DBL:LGW-146/X3H2-97-349 **Language Opportunity:** The concept of substitutability is cerntal to the ADT extension of SQL; currently, pertinent information is scattered over a multitude of subclauses. It needs to be summarized in a separate subclause of the Concepts section. **Solution** None provided with comment.	
	NLD-P02-128		2-Minor Technical	*P02-No specific location*	FND-624 The following Language Opportunity has been noted: **Source:** DBL:LGW-146/X3H2-97-349 **Language Opportunity:** Viewed tables allow the owner of a table to define a subset of its rows and/or columns. The owner may then grant access to the viewed table to other users without giving access to the base table itself. There is no corresponding capability provided with reeference types. To access a column of a row for which a user has a reference, the user is required to have SELECT privilege on the column of the base table. To alter such a column, the user must have UPDATE privilege on the column of the base table. A mechanism analogous to views on base tables is extremely desirable for adequate granualrity of access control.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution None provided with comment.	

	NLD-P02-129		2-Minor Technical	*P02-No specific location*	FND-626 The following Language Opportunity has been noted:
					Source: DBL:LGW-146/X3H2-97-349
					Language Opportunity:
					The <dereference operation> is a very nice syntactic shorthand to avoid the writing of a join. This operation should be extended to allow the use of existing referential constraints.
					`CREATE TABLE enrollments (`
					`student_lname CHAR VARYING (30),`
					`student_fname CHAR VARYING (30),`
					`course REFERENCES courses (id),`
					`grade CHAR VARYING (2),`
					`FOREIGN KEY (student_lname, student_fname) REFERENCES students (lname, fname)`
					`) ;`
					`SELECT course -> course-name,`
					`(student_lname, student_fname) -> address`
					`FROM enrollments`
					`WHERE grade = 'A+' ;`
					Solution
					None provided with comment.	

	NLD-P02-130		2-Minor Technical	*P02-No specific location*	FND-627 The following Language Opportunity has been noted: **Source:** DBL:LGW-146/X3H2-97-349 **Language Opportunity:** A reference type should be able to refer to a cell of a table and not just the entire row. **Solution** None provided with comment.	
	NLD-P02-131		2-Minor Technical	*P02-No specific location*	FND-629 The following Language Opportunity has been noted: **Source:** DBL:LGW-080/X3H2-97-??? **Language Opportunity:** The SQL3 specifications for <attribute definition>, <routine specification>, and <abstract data type body> prohibit the ability to define an explicit mutator function on a single attribute of an ADT with the same signature as the implicit one specified in <attribute definition> (thereby over-riding the implicit one). This sometimes makes it difficult to choose meaningful names both for the attributes of an ADT and for its associated mutator functions. For example, with the comment attribute of the SI_StillImage ADT, it is not possible to define both an attribute name and an explicit mutator function on that attribute with the same name, e.g. COMMENT cannot beused for both names. It is an SQL3 Language Opportunity to provide new syntax in the SQL3 <attribute definition> to allow the implicit mutator function to be explicitly	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					renamed (e.g. similar to the way the CONSTRUCTOR option allows the implicit constructor function of an ADT to be renamed) so that the more desirable attribute name can then be used to define an explicit mutator function for that attribute. Example Usage: <attribute name> <data type> [MUTATOR <mutator name>]. This new syntax might then be used to allow definition of a comment attribute in the SI_StillImage ADT, with its implicit mutator function renamed to be commentOnly, thereby allowing COMMENT to be used as the name of an explicit mutator function that modifies both the comment and the updateTime attributes of the ADT. **Solution** None provided with comment.	
	NLD-P02-132		2-Minor Technical	*P02-No specific location*	FND-630 The following Language Opportunity has been noted: **Source:** DBL:LGW-081/X3H2-97-??? **Language Opportunity:** Would it be possible to allow very limited Type Templates in SQL3 like `DECLARE TYPE TEMPLATE Pixel(`*n*` SMALLINT) AS BIT(`*n*`)` where an upper limit on the value of *n* is implementation-defined, but with the ability to specify the value of *n* as an integer <value expression> whenever Pixel(*n*) is declared as a parameter in an SQL-invoked routine or as an SQL variable in a compound statement. **Solution** None provided with comment.	
	NLD-P02-133		2-Minor Technical	*P02-No specific location*	FND-676 The following Language Opportunity has been noted: **Source:** DBL:LGW-152/X3H2-97-352 (also DBL:LGW-023/X3H2-97-044, SEQ# 469, FRANCE-F-015*) **Language Opportunity:** Some types can be named by themselves (distinct types ADTs and named row types) while others only by defining domains on them (collections row types). This unorthogonality should be removed by allowing any type to be associated to a name through type declaration. **Solution** None provided with comment.	
	NLD-P02-134		2-Minor Technical	*P02-No specific location*	FND-696 The following Language Opportunity has been noted: **Source:** DBL:BBN-128/X3H2-98-354 (BBN-029R1, SEQ#149, USA-P02-034) **Language Opportunity:** The restriction that only rows of persistent base tables can be referenced should be lifted to allow references to nested (un-named) row types. **Solution** None provided with comment.	
	NLD-P02-135		2-Minor	*P02-No specific location*	FND-707 The following Language Opportunity has been noted:	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
			Technical		**Source:** Email from Mike Ubell 5 August, 1998 **Language Opportunity:** In X3H2-98-016, the ability to dynamically dispatch a function was eliminated in favor of method based dispatch. This was done to bring SQL more in line with Java and therefore, presumably, make it easier to import non-SQL written shrink wrap applications into the database. Unfortunately many existing applications (and data type packages) are not written in Java today, or even in C++. By removing the multi-argument dispatch data types that support comparison and inheritance must dispatch on one argument within the method. If the method is implemented in a language that does not support inheritance, then new subtypes may not be added to the shrink-wrapped data type. **Solution** None provided with comment.	
	NLD-P02-136		2-Minor Technical	*P02-No specific location*	FND-719 The following Language Opportunity has been noted: **Source:** WG3:YGJ-021 **Language Opportunity:** The reference type and the dereference operator have been added to SQL3. The ability to update a column or delete a row via a reference must be supplied as well. **Solution** None provided with comment.	
	NLD-P02-137		2-Minor Technical	*P02-No specific location*	FND-720 The following Language Opportunity has been noted: **Source:** WG3:YGJ-021 **Language Opportunity:** SQL3 requires that a table have an associated user-defined type in order to be referenceable. The combination of user-defined type and base table is now very difficult to change in any way. The two would have to be disassociated, each altered separately, and then associated again. Neither the disassociation of user-defined type and base table nor the altering of a user-defined type are supported. **Solution** None provided with comment.	
	NLD-P02-138		2-Minor Technical	*P02-No specific location*	FND-721 The following Language Opportunity has been noted: **Source:** WG3:YGJ-021 **Language Opportunity:** Constraints are not a part of a user-defined type. This means that the constraints that are intended for each table of a user-defined type must be explicitly copied and maintained by a user. **Solution** None provided with comment.	
	NLD-P02-139		2-Minor Technical	*P02-No specific location*	FND-722 The following Language Opportunity has been noted: **Source:** WG3:FRA-092R2	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Language Opportunity: The table defining features in Core SQL should be examined to ensure that the features exhaust all ov Core (perhaps by showing that all BNF nonterminals that are available to Core have been assigned to some faeture) and that they are rigorously stated. **Solution** None provided with comment.	
	NLD-P02-140		2-Minor Technical	*P02-No specific location*	FND-758 The following Language Opportunity has been noted: **Source:** WG3:BHX-149 **Language Opportunity:** If might be useful to add to SQL the ability to use explicit character set names taken from the public registry for character set names (an IANA [Internet Assigned Numbers Authority] registry available at `ftp://ftp.isi.edu.in-notes/iana/assignments/character-sets`). **Solution** None provided with comment.	
	NLD-P02-141		2-Minor Technical	*P02-No specific location*	FND-773 The following Language Opportunity has been noted: **Source:** WG3:BHX-107/H2-2000-___ **Language Opportunity:** It is desirable to provide the capability on CREATE TABLE to change options (scope, reference checking, NOT NULL specification, default values, datalink control definitions, and so on) that are associated with components nested inside row types, collection types, and structured types. **Solution** None provided with comment.	
	NLD-P02-142		2-Minor Technical	*P02-No specific location*	FND-778 The following Language Opportunity has been noted: **Source:** WG3:BHX-117/H2-2000-___ **Language Opportunity:** WG3:SLD-046 added several new fields to the CLI descriptor area: CURRENT_TRANSFORM_GROUP, SPECIFIC_TYPE_CATALOG, SPECIFIC_TYPE_SCHEMA, and SPECIFIC_TYPE_NAME. The same fields could profitably be added to the SQL descriptor area, too. **Solution** None provided with comment.	
	NLD-P02-143		2-Minor Technical	*P02-No specific location*	FND-780 The following Language Opportunity has been noted: **Source:** WG3:HEL-047/H2-2000-___ **Language Opportunity:** 2. Insurmountable (?) problem for query generators The unfriendliness described in FND-779 causes a certain difficulty to general purpose applications, such as query generators, that appears to be insurmountable. Given two arbitrary character string expressions of character set	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					CS, there is no guaranteed way of having them compared under the default collation of CS without knowing what that collation is. Moreover, the default collation can be looked up in the Information Schema only if the character set CS itself is known. There is no sure way that we are aware of whereby the character set of an arbitrary string expression can be determined by an SQL application. **Solution** None provided with comment.	
	NLD-P02-144		2-Minor Technical	*P02-No specific location*	FND-787 The following Language Opportunity has been noted: **Source:** WG3:PER-146/H2-2001-??? (FCD1/2000 WG3-P01-011) **Language Opportunity:** [Jake Knoppers] saw that with respect to "normative references" point 1p that ISO 8601:2001 version is to be referenced. This is good; [he works] on that standard. [His]comment is that serious consideration should also be given to referencing ISO 19108:2000 "Geographic information - - Temporal schema". ISO 8601 deals mainly with Gregorian calendar referencing. Increasingly, various areas of business application such as banking/financial services, geomatics, intelligent transportation systems, etc. use other calendar referencing systems, such as the GPS clock, which is use for synchronization among the global position satellites and provides for a "common" single world wide date/time referencing among IT systems of autonomous organizations (one then maps the GPS date/time stamp to one's local time, whatever it is). It is likely that many SQL based implementations will do the same. [He does] not know whether you want to treat this as a "comment" an "informative note/footnote", etc. but [he thinks] that it is important for SQL users. **Solution** None provided with comment.	
	NLD-P02-145		2-Minor Technical	*P02-No specific location*	FND-788 The following Language Opportunity has been noted: **Source:** WG3:PER-146/H2-2001-??? (FCD1/2000 WG3-P01-018) **Language Opportunity:** Allow implementations to be able to represent year numbers outside of 0001-9999 (0000 is 1 B.C, etc.). The restriction of YEAR to be between 0001 and 9999 is unsupportable. Note also that ISO/IEC 8601:2001 does not have any such restriction; 0000 and negative years are allowed (year 0000 is year 1 BC, -0001 is year 2 BC, ...), as are year indications with more than 4 digits. Further, sub-second precision should be possible to use (i.e. required by the standard). (Note: The CD Editing Meeting believes that this sentence means that implementations should be mandated to supply significant digits, other than zero, to the right of the decimal point, although there may be hardware that does not support "clock ticks" at such a fine granularity.) **Solution**	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					None provided with comment.	
	NLD-P02-146		2-Minor Technical	*P02-No specific location*	FND-789 The following Language Opportunity has been noted: **Source:** WG3:PER-146/H2-2001-??? (FCD1/2000 WG3-P02-010) **Language Opportunity:** Allow decimal numbers to be expressed using any one (for each numeral) of the decimal number category (Nd) ranges in the UCS. Conversely, there should also be a way of getting out formatted numbers using a specified range (by script name or similar) of Nd characters. Allow the character MINUS as an 'alias' to HYPHEN-MINUS in arithmetic expressions. Allow LESSTHAN OR EQUAL, GREATER-THAN OR EQUAL, as well as LESS-THAN OR SLANTED EQUAL (Unicode 3.2), and GREATER-THAN OR SLANTED EQUAL (Unicode 3.2) with their obvious comparison semantics. Allow DOT OPERATOR for multiplication. **Solution** None provided with comment.	
	NLD-P02-147		2-Minor Technical	*P02-No specific location*	FND-791 The following Language Opportunity has been noted: **Source:** WG3:PER-146/H2-2001-??? (FCD1/2000 USA-P02-010) **Language Opportunity:** There is no discussion of the relationship between determinism and isolation level. Two read transactions starting at the exact same time working on the "same" SQL data can still have different results if they operate on different isolation levels. The May, 2001 CD Editing Meeting in Perth observed that describing such interactions is extremely difficult and all such descriptions known to the Editing Meeting participants rely heavily (perhaps exclusively) on the locking paradigm, which the standard does not require. Because of this, the Editing Meeting believed that a complete resolution of this Language Opportunity is quite unlikely. **Solution** None provided with comment.	
	NLD-P02-148		2-Minor Technical	*P02-No specific location*	FND-807 The following Language Opportunity has been noted: **Source:** WG3:PER-171/H2-2001-??? (FCD1/2000 USA-P02-010) **Language Opportunity:** It may be useful to have a notion of "hereditary property" of BNF nonterminals. A hereditary property P would work like this: If A ::= B, then P(A) = P(B), unless there is an explicit syntax rule to the contrary. Examples of hereditary properties would be declared type, scale, precision, most specific type, value. This is already the haphazard approach of the standard, for example, to say in one SR that "the data type of B is DT" and then later assume that the data type of A is DT since A ::= B. **Solution**	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					None provided with comment.	
	NLD-P02-149		2-Minor Technical	*P02-No specific location*	FND-827 The following Language Opportunity has been noted: **Source:** FCD1/2000 WG4-P02-001 **Language Opportunity:** It should be allowed to invoke a method using a <routine invocation> with a signature that is identical to the <method selection> specified in Subclause 6.16, "<method invocation>", and in Subclause 6.17, "<static method invocation>", respectively. **Solution** None provided with comment.	
	NLD-P02-150		2-Minor Technical	*P02-No specific location*	FND-830 The following Language Opportunity has been noted: **Source:** WG3:PER-188/H2-2001-??? **Language Opportunity:** In the mathematical community, multiset union of *M1* and *M2* is defined as consisting of every element that is an element of either *M1* or of *M2*, occurring either as many times as it does in *M1* or as many times as it does in *M2*, whichever is the greater. (The SQL operator called UNION ALL, and also called MULTISET UNION after acceptance of WG3:PER-098 is referred to as "union plus", denoted thus: U+.) The mathematical definition of multiset union seems just as good a counterpart of the multiset intersection we already have as union plus does, because intersection can be expressed by just changing "either" to "both", "or" to "and", and "greater" to "lesser" in the above informal definition of multiset union. **Solution** None provided with comment.	
	NLD-P02-151		2-Minor Technical	*P02-No specific location*	FND-831 The following Language Opportunity has been noted: **Source:** The merger of X3H2-95-178/DBL:YOW-048, X3H2-95-201/DBL:YOW-049R, and X3H2-95-179R2/DBL:YOW-050R (Was Language Opportunity PSM-061) **Language Opportunity:** Exceptions that are passed back through a routine invocation should be traceable. The list of <routine invocations> that they were propagated back through should be made available somewhere, such as in the Diagnostics Area. **Solution** None provided with comment.	
	NLD-P02-152		2-Minor Technical	*P02-No specific location*	FND-848 The following Language Opportunity has been noted: **Source:** WG3:YYJ-016 (USA-P02-113) **Language Opportunity:** A number of DBMS products support materialized views whose results are stored in the database and subsequently maintained by the system whenever any of the generally underlying base tables of the views changes. Materialized views play an important role in offering significant performance gains for complex	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					queries, especially in Data Warehouse applications. The next edition of the SQL standard should standardize the syntax and semantics of materialized views. **Solution** None provided with comment.	
	NLD-P02-153		2-Minor Technical	*P02-No specific location*	FND-849 The following Language Opportunity has been noted: **Source:** WG3:YYJ-016 (USA-P02-114) **Language Opportunity:** In [FoundationCD], it is possible write insert and update statements where the value of one or more fields are not immediately known by the updater. This includes columns populated by subqueries, functions, system values, etc. In some cases, the updater needs to know the values after the insert/update has occurred. In some cases, this can be accomplished by requerying the data after the update. In other cases, the updater cannot easily requery the data. This includes cases such as when a function is used to generate the primary key. For example: Insert into T1 (c1 , c2 , c3) values (fn_generate_pk('T1') , :var 2 , :var 3); It would be useful to have a mechanism that allows an insert or update statement to return the inserted or updated rows in a singleton select or a cursor. **Solution** None provided with comment.	
	NLD-P02-154		2-Minor Technical	*P02-No specific location*	FND-850 The following Language Opportunity has been noted: **Source:** WG3:YYJ-016 (USA-P02-117) **Language Opportunity:** SQL should be enhanced to support EJB Query Language. Information about the EJB Query Language can be found the public document available at: http://java.sun.com/aboutJava/communityprocess/first/jsr019/ejb2-finaldraft.pdf particularly in Chapter 10. **Solution** None provided with comment.	
	NLD-P02-155		2-Minor Technical	*P02-No specific location*	FND-876 The following Language Opportunity has been noted: **Source:** WG3:DRS-128 **Language Opportunity:** SQL/Foundation, as currently written, prohibits the creation and invocation of multiple polymorphic routines whose parameters differ only by character set or by interval class (year-month or day-time). This is clearly unacceptable for many users' needs. This Opportunity has been "narrowed" by acceptance of WG3:FRA-120R1. It was formerly PSM-127 . **Solution** None provided with comment.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
	NLD-P02-156		2-Minor Technical	*P02-No specific location*	FND-913 The following Language Opportunity has been noted: **Source:** WG3:ZSH-155 = H2-2002-___ **Language Opportunity:** There should be an explicit specification of what features a conforming Syntax Only SQL Flagger must detect. **Solution** None provided with comment.	
	NLD-P02-157		2-Minor Technical	*P02-No specific location*	FND-914 The following Language Opportunity has been noted: **Source:** WG3:ZSH-155 = H2-2002-___ **Language Opportunity:** Suppose you have defined a structured UDT with 50 attributes. In order to grant somebody else the right to retrieve and set the values of each of those attributes, you must execute no fewer than 101 GRANT statements! First, you must grant USAGE on the type itself. Then, you must grant EXECUTE on each of the 50 observer methods and EXECUTE on each of the 50 mutator methods. The process is particularly cumbersome, because granting EXECUTE on the observer methods requires something like "GRANT EXECUTE ON INSTANCE METHOD attribute_n FOR typename TO username" (which is easy enough), but granting EXECUTE on the mutator methods requires something like "GRANT EXECUTE ON INSTANCE METHOD attribute_n (argument-type-1, argument-type-2,...argument-type-n) FOR typename TO username". Of course, you could choose to use the <specific name> for the methods, but those names are likely to be awkward and/or non-intuitive. The process of entering all of those GRANTs is incredibly unfriendly to type definers and grows worse as UDTs get more complex. Contrast this with the process of granting retrieval and modification privileges on a table with 1000 columns: "GRANT SELECT ON tablename TO username" and "GRANT UPDATE ON tablename TO username". That's it. Granting (and revoking!) access privileges to attributes of UDTs should be made more user-friendly. **Solution** None provided with comment.	
	NLD-P02-158		2-Minor Technical	*P02-No specific location*	FND-915 The following Language Opportunity has been noted: **Source:** WG3:ZSH-155 = H2-2002-___ **Language Opportunity:** Instead of trying to discover and remember all the possible dependencies between schema objects, what we should do is create the dependency at the time of creating the dependent object. This should enable a simplification of the rules for DROP and REVOKE, as well as making them more intelligible and easier to maintain. **Solution** None provided with comment.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
	NLD-P02-159		2-Minor Technical	*P02-No specific location*	FND-916 The following Language Opportunity has been noted: **Source:** WG3:ZSH-155 = H2-2002-___ **Language Opportunity:** The character string "associate" occurs 373 times in [FoundFCD], mostly in the phrase "associated with". In many cases the meaning, or effect, of an association between two objects can be found only by finding all the places where it is mentioned. In a number of such cases the phrase could be avoided altogether, in others the significance of the association could be more explicitly explained. We give one or two examples where it does not appear difficult to avoid the phrase. Subclause 03.03.01.01, "Other terms", ... <SQL statement variable> that was associated with an <SQL statement name> by a <prepare statement> ... Subclause 04.02.01, "Character strings and collating sequences", Each collation known in an SQL-environment is applicable to one or more character sets, and for each character set, one or more collations are applicable to it, one of which is **associated with it as** its character set collation. The words in bold are unnecessary, and could well be deleted altogether. The word "default" could be added, between "its" and "character set". Subclause 05.04, "Names and identifiers", Syntax Rule 17) 17) An <identifier> that is a <correlation name> **is associated with** a table within a particular scope. The scope of a <correlation name> is either a <select statement: single row>, <subquery>, or <query specification> (see Subclause 7.6, "<table reference>"), or is a <trigger definition> (see Subclause 11.39, "<trigger definition>"). Scopes may be nested. In different scopes, the same <correlation name> **may be associated with** different tables or with the same table. **Solution** None provided with comment.	
	NLD-P02-160		2-Minor Technical	*P02-No specific location*	FND-917 The following Language Opportunity has been noted: **Source:** WG3:ZSH-153R1 = H2-2002-153R1 **Language Opportunity:** The concepts section needs to explain that CAST AS is the mechanism to translate datetime and interval data types to and from host data parameters. **Solution** None provided with comment.	
	NLD-P02-161		1-Major Technical	*P02-No specific location*	FND-974 The following Possible Problem has been noted: **Source:** WG3:SIA-031 = H2-2004-??? **Possible Problem:** The General Rules applying to <rollback statement> are incomplete, and inconsistent with the text of Subclause 4.35.2, “Savepoints”. General Rule 2) is, in part:	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					1) If a <savepoint clause> is not specified, then: ... a) Every valid locator is marked invalid. b) All open cursors in any SQL-client module associated with the current SQLtransaction are closed. Nothing is said about locators or cursors held from the previous transaction. Nor is anything said about prepared statements. General Rule 3) is, in part: 1) If a <savepoint clause> is specified, then: ... a) Every valid locator that was generated in the current SQL-transaction subsequent to the establishment of S is marked invalid. b) For every open cursor CR in any SQL-client module associated with the current SQL-transaction that was opened subsequent to the establishment of S, the following statement is implicitly executed: `CLOSE CR` c) The status of any open cursors in any SQL-client module associated with the current SQL-transaction that were opened by the current SQL-transaction before the establishment of S is implementation-defined. NOTE 497 — The current SQL-transaction is not terminated, and there is no other effect on the SQL-data or schemas. This General Rule says nothing corresponding to the content of the final paragraph of Subclause 4.35.2, "Savepoints", nor about restoring some settable elements of the SQL-session context, e.g. current role name. **Solution:** Specify what happens in terms of the contents of the SQL-session context. **Solution** None provided with comment.	
	NLD-P02-162		2-Minor Technical	*P02-No specific location*	FND-980 The following Language Opportunity has been noted: **Source:** WG3:STX-020 **Language Opportunity:** A <set role statement> always raises an exception if there is no current user identifier. This prevents the use of definer's rights routines (where the definer is a role) to handle the setting of roles. If this is desired functionality, then one of the following alternatives should be chosen and implemented. a) Allow a <set role statement> if there is a current user identifier and the role is an applicable role for that user identifier or, if there is no current user identifier, then if the role is an applicable role for the SQL-session user identifier. b) Allow a <set role statement> if the role is an applicable role for the current user identifier or the SQLsession user identifier. c) Allow a <set role statement> if the role is an applicable role for the most rhecent current user identifier. That is the user identifier with the highest stack	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					position. **Solution** None provided with comment.	
	NLD-P02-163		1-Major Technical	*P02-No specific location*	**NLD-P02-052** When the General Rules of another Subclause are invoked the specification of the paramater passing is not always correct. Sometimes the name of the argument(s) is(are) not explicitly given and sometimes the arguments are not correctly identified. In the latter case is is sometimes because the invoked Subclause does not itself given clearly identifiable names to its arguments. All the calling and called Subclauses should be checked and corrected. See also FND-948 **Solution** None provided with comment.	
	NLD-P02-164		3-Minor Technical	*P02-No specific location*	**WG3-P02-006** A (so far unknown) possible problem is identified and resolved. Currently, certain DDL statements do not have a restriction to disallow a <host parameter specification>, an <SQL parameter reference>, a <dynamic parameter specification>, or an <embedded variable specification> (and, a <SQL variable reference> in PSM) in their definitions. **Solution** None provided with comment.	
SQL/CLI						
	NLD-P03-001		1-Major Technical	*P03-05.04, CLI Implicit Cursor*	**WG3-P03-001** GR 8) b) "The General Rules of Subclause 14.1, "<declare cursor>", in ISO/IEC 9075-2 are applied to CR." Doesn't work because it doesn't say what CR is equivalent to in <declare cursor>, and in any case I don't think there is anything to be equivalent to - CR is not defined in <declare cursor> only in <open cursor>. **Solution** None provided with comment.	
	NLD-P03-002		1-Major Technical	*P03-05.06, Implicit EXECUTE USING and OPEN USING clauses*	**WG3-P03-002** In GR 4) p) 2) 1) B) II) "If the <cast specification> CAST (*SV* AS *TDT*) does not conform to the General Rules of Subclause 6.12, "<cast specification>", in ISO/IEC 9075-2, then an exception condition is raised in accordance with the General Rules of Subclause 6.12, "<cast specification>", in ISO/IEC 9075-2. I don't think you can conform to "General Rules". Also why is this rule not covered by the following subrule (III) which	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					(effectively) invokes the syntax and General Rules of 6.12, "<cast specification>". **Solution** None provided with comment.	
	NLD-P03-003		1-Major Technical	*P03-06.19, ExecDirect*	**WG3-P03-003** This subclause needs to be examined to see if it needs similar treatment to that proposed in SIA-029 (and accepted) for Foundation, Subclause 19.12, "<execute statement>". A similar comment applies to Subclause 6.20, "Execute". The drafting of this P.P. completes Action Item k), recorded in STX-012, "Action Items from Xi'an". **Solution** None provided with comment.	
	NLD-P03-004		4-Minor Editorial	*P03-06.33 GetDiagField*	CLI-053 The following Possible Problem has been noted: **Source:** WG3:HBA-042 = H2-2003-___ **Possible Problem:** Hanging between In General Rule 12) i), hanging between Subrules ii) C) and iii), is the sentence: If the value of TABLE_NAME identifies a declared local temporary table, then the value of CATALOG_NAME is <space>s and the value of SCHEMA_NAME is 'MODULE'. The second sentence of General Rule 12) i) iii) 1) is: If TABLE_NAME refers to a declared local temporary table, then CATALOG_NAME is <space>s and SCHEMA_NAME contains 'MODULE'. It rather looks as though the former was intended to replace the latter, since the style of wording it uses seems to be more prevalent. SQL:1999 contains the same error. Solution Replace the second sentence of General Rule 12) i) iii) 1) (as quoted above) with: If the value of TABLE_NAME identifies a declared local temporary table, then the value of CATALOG_NAME is <space>s and the value of SCHEMA_NAME is 'MODULE'. **Solution** None provided with comment.	
	NLD-P03-005		1-Major Technical	*P03-No specific location*	**NLD-P03-002** When the General Rules of another Subclause are invoked the specification of the paramater passing is not always correct. Soemtimes the name of the argument(s) is(are) not explicitly given and sometimes the arguments are not correctly identified. In the latter case is is sometimes because the invoked Subclause does not itself given clearly identifiable names to its arguments. All the calling and called Subclauses should be checked and corrected.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Solution None provided with comment.	
	NLD-P03-006		2-Minor Technical	*P03-06.17, EndTran*	CLI-055 The following Possible Problem has been noted: **Source:** WG3:STX-053 Addressing Action Item n) from Xian on SIA031 **Possible Problem:** SIA-031 Para 2.1.1 made Changes to Working Draft SQL/Foundation Subclause 16.7, "<commit statement>" This change should be made to the COMMIT rules in EndTran. **Solution** None provided with comment.	
	NLD-P03-007		2-Minor Technical	*P03-04.03, Diagnostics areas in SQL/CLI and P03-0A.2, COBOL library item SQLCLI*	CLI-056 The following Possible Problem has been noted: **Source:** WG3:STX-053 Addressing Action Item n) from Xian on SIA031 **Possible Problem:** SIA-031 Para 2.2.3 made Changes to SQL/Foundation Annex B, "Implementation-defined elements" The diagnostics areas in CLI are different than those of embedded/module SQL. But there does not appear to be any indication in CLI of whether it covers the relationship of the CLI diagnostics area to the impact of a ROLLBACK to a SAVEPOINT. We may need to modify CLI to make the impact on the CLI diagnostics area implementation-defined but before doing this we might want to check what CLI implementations do today. **Solution** None provided with comment.	
	NLD-P03-008		2-Minor Technical	*P03-05.04, Implicit cursor*	CLI-013 The following Language Opportunity has been noted: **Source:** X3H2-98-077R1/DBL:BBN-??? and Paul Cotton, March 1, 1998 **Language Opportunity:** General Rule 7)e) "Case" i) "If CR specifies INSENSITIVE..." carries out the same functionality as expressed in the General Rules of SQL/Foundation <open cursor>. It is a Language Opportunity to reference the appropriate rules in SQL/Foundation instead of repeating them here. **Solution** None provided with comment.	
	NLD-P03-009		2-Minor Technical	*P03-08.01, Claims of conformance to SQL/CLI*	CLI-026 The following Language Opportunity has been noted: **Source:** X3H2-98-077R1/DBL:BBN-??? and Source: Paul Cotton, March 1, 1998 **Language Opportunity:** Would it make sense to have a CLI flagger which discovers nonportable extensions? One way to do this would be to set an environment attribute (if there is such a thing) saying that any use of a nonportable argument will return a special status code. CLI should support this requirement only if it is also	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					required for conformance to dynamic SQL. **Solution** None provided with comment.	
	NLD-P03-010		2-Minor Technical	*P03-No specific location*	CLI-047 The following Language Opportunity has been noted: **Source:** FCD (1999) ballot comment USA-P03-024 **Language Opportunity:** WG3:SLD-010/X3H2-98-027R3 provides for fetching multiple rows in one CLI routine invocation. It would be appropriate to be able to provide an array of input parameter values to a single statement execution in a similar fashion. **Solution** None provided with comment.	
	NLD-P03-011		2-Minor Technical	*P03-No specific location*	CLI-048 The following Language Opportunity has been noted: **Source:** FCD (1999) ballot comment USA-P03-025 **Language Opportunity:** WG3:SLD-010/X3H2-98-027R3 provides for fetching multiple rows into an array of variables. CLI should also be able to specify 'row-wise binding', so that the application can bind to an array of record structures, where fields of the record structure are the input or output parameters. **Solution** None provided with comment.	
	NLD-P03-012		2-Minor Technical	*P03-06.21, Fetch*	CLI-049 The following Language Opportunity has been noted: **Source:** WG3:SLD-010/X3H2-99-027R3 **Language Opportunity:** The arrays that receive the results from multi-row fetches must be contiguous. For example, if you are performing `SELECT EMPNO, NAME FROM EMP` the application cannot create a record structure with fields for EMPNO and NAME, and then create an array of these structures. The reason is that all the EMPNOs will be delivered in a single contiguous array, and all of the NAMEs will be delivered in a separate array. It would be useful to provide for an offset with a record structure or a "stride" (distance between successive elements of an array). This is a method of binding known as row-wise binding. Row-wise binding was deliberately not part of the paper that proposed multi-row fetch since it is an orthogonal enhancement and therefore benefits by being considered in a separate proposal. We note in passing that row-wise binding can be accomplished simply and elegantly by introducing a new descriptor field that informs whether the buffers are laid out as 'regular' (or column-wise) binding, or as row-wise binding. **Solution** None provided with comment.	
	NLD-P03-013		2-Minor Technical	*P03-06.34, GetDiagRec*	CLI-054 The following Language Opportunity has been noted: **Source:** WG3:STX-001 Action Item n) Mark Ashcroft. Additional to SIA-	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					025R1 **Language Opportunity:** SIA-025R1, "A Shorthand for Getting ALL Diagnostics" proposes to add a new diagnostics option to embedded SQL but ignores the question of whether it should also be added to CLI or the SQLJ binding. **Solution** None provided with comment.	
SQL/PSM						
	NLD-P04-001		1-Major Technical	*P04-05.02, Names and identifiers*	PSM-153 The following Possible Problem has been noted: **Source:** WG3:HBA-042 = H2-2003-___ **Possible Problem:** SQL/Foundation, Subclause 5.4, “Names and identifiers”, Syntax Rule 5) a) says: a) If LSQ [a <local or schema qualifier>] is "MODULE", then TN [a <table name>] shall be contained in an <SQL-client module definition> M and the <module contents> of M shall contain a <temporary table declaration> TT whose <table name> has a <qualified identifier> equivalent to QI. This does not cater for the case of a <temporary table declaration> referenced by a <table name> contained in a <module routine>. **Solution** Perhaps PSM should replace the cited Syntax Rule 5) a) with: a) If LSQ [a <local or schema qualifier>] is “MODULE”, then TN [a <table name>] shall be contained either in an <SQL-client module definition>, without an intervening <SQL-schema statement>, or in a <SQL-client module definition> that contains a <temporary table declaration> TT whose <table name> has a <qualified identifier> equivalent to QI.	See Comment
	NLD-P04-002		1-Major Technical	*P04-05.02, Names and identifiers*	PSM-154 The following Possible Problem has been noted: **Source:** WG3:HBA-042 = H2-2003-___ **Possible Problem:** Whatever is said, in the Syntax or General Rules of SQL/Foundation, Subclause 5.4, “Names and identifiers”, about how a <table name> identifies a created local temporary table must be replaced by PSM to cater for <module routine>s. Let TT be a local created temporary table; let R1 be SQL-routine in SQL-server module M1 and let R2 be SQL-routine in SQL-server module M2. Both R1 and R2 contain references to TT. It is understood that, regardless of the pattern of invocation, each of R1 and R2 has its own instance of TT. None of the foregoing is specified in any General Rule, and is described inadequately Concepts. It needs to be properly specified. **Solution** None provided with comment.	
	NLD-P04-003		1-Major	*P04-08.01,*	PSM-155 The following Possible Problem has been noted:	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
			Technical	*<routine invocation>*	**Source:** WG3:HBA-042 = H2-2003-___ **Possible Problem:** SQL/Foundation, Subclause 10.4, "<routine invocation>", General Rule 5) d) i) is: i) If R is an SQL routine, then remove from RSC the identities of all instances of created local temporary tables, ... This doesn't look good for an SQL routine R1 contained in an SQL-server module, that might invoke another <module routine> R2 in the same module. **Solution:** PSM must modify the cited subrule in some way. **Solution** None provided with comment.	
	NLD-P04-004		1-Major Technical	*P04-08.01, <routine invocation>*	PSM-156 The following Possible Problem has been noted: **Source:** WG3:HBA-042 = H2-2003-___ **Possible Problem:** It is not clear whether the rows of a temporary table (whether declared or created) that is local to an SQLserver module survive from one invocation of SQL-routines in that module. Let M be a SQL-server module and TT either a declared temporary table local to it, or a local created temporary table; let R be an SQL-routine in M that accesses TT. R is invoked twice during a transaction from by some invoker INV. It seems to be intended that, provided INV does not, between the invocations of R1, access TT, the second invocation of R will find TT as the first invocation left it. Furthermore, if INV is an SQL-routine in M, and INV accesses TT, then INV and R access the same (instance of) TT. On the other hand, if INV is in an SQL-server module MI, different from M, then, whether TT is a declared temporary table or a local created temporary table, a reference to TT in INV, if valid, identifies a different table from the one accessed by R. Then again, if R invokes a further SQL-routine RS, that, like INV, is in MI, then RS will see the same TT as INV. The foregoing is not specified in any General Rule, nor is it clearly described in Concepts. **Solution** None provided with comment.	
	NLD-P04-005		1-Major Technical	*P04-10.03, <revoke statement>*	PSM-149 The following Possible Problem has been noted: **Source:** CD1-2000 comments USA-P04-005 **Possible Problem:** Because PSM expands the possibilities of <SQL procedure statement>, the capabilities for the <triggered action> of a trigger are much increased. Consequently the rules regarding dependencies of a trigger on a privilege or	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					schema object must be extended in PSM. For example, in 9075-2 (SQL/Foundation), Subclause 12.7, "<revoke statement>", SR 24) subrules g) through j) deal with when SELECT privilege is required to define a trigger. None of these rules cover the possibility of a <scalar subquery> in a <case statement>. Likewise the rules for SELECT WITH HIERARCHY OPTION are inadequate. The commenter does not believe that the solution is to run around trying to find every case that is not currently covered. Instead, the commenter believes that we need a general mechanism that constructs a dependency graph relating arbitrary schema objects and privileges, so that as features and parts are added, each new feature or part need only specify its contribution to the dependency graph algorithm. For example, dependencies on privileges can be declared in the Access Rules, so that whenever an Access Rule is used, a dependency is automatically created. That way <revoke statement> would not need to duplicate information that is already implicit in the Access Rules. Similarly, dependencies on schema objects can be generated in the rules of <table reference>, <column reference>, etc. Then <revoke statement> and the drop statements would not need to generate dependencies, they could simply assume that they are defined. **Solution** None provided with comment.	
	NLD-P04-006		1-Major Technical	*P04-11.02, <SQL procedure statement>*	PSM-158 The following Possible Problem has been noted: **Source:** DCOR/2004 WG3-P04-001 **Possible Problem:** General Rule 3) calls for the General Rules of , "<handler declaration>" to be applied. However, it does not say for which handler the rules should be applied and there does not seem to be any clear context available in either <SQL procedure statement> or in <handler declaration> to make an implicit choice. **Solution** None provided with comment.	
	NLD-P04-007		1-Major Technical	*P04-No specific location*	PSM-152 The following Language Opportunity has been noted: **Source:** WG3:HBA-040 **Language Opportunity:** The scope of an extended name that contains LOCAL is not adequately specified in the following cases: a) Where the extended name is contained in an <SQL control statement> immediately contained in an <externally-invoked procedure>. b) Where the extended name is contained in a <schema routine>. c) Where the extended name is contained in a <module routine> The determination of what object, if any, is identified by an extended name should not depend on the statement that contains it being contained in a particular <SQL-server module definition>, still less a particular <SQL-client module definition>.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					For consistency, if a local scope is to be permitted in these cases, it should follow the precedent of <cursor name>, provided <statement name> also follows it. The preferred solution, however, is to make all extended names global, by deleting <scope option>. **Solution** None provided with comment.	
	NLD-P04-008		2-Minor Technical	*P04-No specific location*	PSM-088 The following Language Opportunity has been noted: **Source:** Steve Cannan, during the course of discussing DBL:MCI-060 **Language Opportunity:** Need some syntax to do an ALTER VIEW or similar to "rebind" subject routines, * column references, etc. for all objects that contain statically-bound references of any sort. **Solution** None provided with comment.	
	NLD-P04-009		2-Minor Technical	*P04-No specific location*	PSM-095 The following Language Opportunity has been noted: **Source:** Ed Dee, while discussing DBL:MCI-132 ballot comments **Language Opportunity:** FOR statements terminate (with a closed cursor exception) if the statement list of the <for statement> list contains a COMMIT or ROLLBACK. Further, no statement contained in the <for statement> can set any transaction attributes. It is desirable that an application programmer be able to initiate or terminate transactions within a <for statement>. **Solution** None provided with comment.	
	NLD-P04-010		2-Minor Technical	*P04-08.01, <routine invocation>*	PSM-106 The following Language Opportunity has been noted: **Source:** DBL:MCI-161, point 2.5, item 5 **Language Opportunity:** In Subclause 8.1, "<routine invocation>", the prohibitions on SQL-transaction staements and SQL-connection statements in SQL-invoked routines might be lifted, if a way can be found to make sure that SQLinvoked routines end SQL-sessions and SQL-transactions that they start, don't end SQL-transactions and SQL-sessions that they didn't start, and don't switch SQL-connections without restoring the SQL-connection with which they started. **Solution** None provided with comment.	
	NLD-P04-011		2-Minor Technical	*P04-08.01, <routine invocation>*	PSM-107 The following Language Opportunity has been noted: **Source:** Discussion of DBL:MCI-161, point 2.5, item 5 **Language Opportunity:** In Subclause 8.1, "<routine invocation>", the prohibitions on SQL-transaction statements and SQL-connection statements in SQL-invoked routines might be	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					lifted by changing "SQL-connection statement" to "SQL-connection statement and the implementation does not support the execution of that SQL-statement in an invoked SQL-routine that is a procedure" in each of the two rules that make this prohibition, and making an appropriate entry in Annex B, "Implementation-defined elements", saying something like "It is implementation-defined whether or not an SQL-implementation supports the execution of SQL-transaction statements and/or SQL-connection statements in an invoked SQL-routine; if it does so, then the effects are implementation-defined." **Solution** None provided with comment.	
	NLD-P04-012		2-Minor Technical	*P04-No specific location*	PSM-124 The following Language Opportunity has been noted: **Source:** DBL:MCI-040/X3H2-96-169:UK-017 **Language Opportunity:** No way of obtaining the associated sqlstate of a condition name. We think the <condition name> feature is a nice idea, but we suspect it will generate a requirement, akin to the observation in the preceding comment, for a built-in function to return the associated sqlstate value of a given condition name. Furthermore, it might even be required to hold condition names in variables or arguments, in which case they have to become character strings. We would be happy to hold this feature over for SQL3, in the interests of simplification and early progression of PSM2 and to give time for the requirements to be fully thought through and appropriately addressed in the language. **Solution** None provided with comment.	
	NLD-P04-013		2-Minor Technical	*P04-No specific location*	PSM-140 The following Language Opportunity has been noted: **Source:** DBL:LGW-081/X3H2-97-??? **Language Opportunity:** Is it possible in SQL3 to relax the specification of string data types such as <character stringtype> and <bit string type> so that the declared length of these types (with appropriate usage restrictions) can be specified at execution time rather than at compile time? Can I declare avariable in an outer block of a compound statement and then use that variable as the <length> of a bit string variable declaration in an inner block? **Solution** None provided with comment.	
	NLD-P04-014		2-Minor Technical	*P04-No specific location*	PSM-152 The following Language Opportunity has been noted: **Source:** WG3:HBA-040 **Language Opportunity:** The scope of an extended name that contains LOCAL is not adequately specified in the following cases: a) Where the extended name is contained in an <SQL control statement> immediately contained in an <externally-invoked procedure>.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					b) Where the extended name is contained in a <schema routine>. c) Where the extended name is contained in a <module routine> The determination of what object, if any, is identified by an extended name should not depend on the statement that contains it being contained in a particular <SQL-server module definition>, still less a particular <SQL-client module definition>. For consistency, if a local scope is to be permitted in these cases, it should follow the precedent of <cursor name>, provided <statement name> also follows it. The preferred solution, however, is to make all extended names global, by deleting <scope option>. **Solution** None provided with comment.	
	NLD-P04-015		2-Minor Technical	*P04-14, Dynamic SQL*	PSM-157 The following Language Opportunity has been noted: **Source:** WG3:HBA-048 = H2-2003-___ **Language Opportunity:** SQL/Foundation, Subclause 19.10, "<input using clause>", Syntax Rule 1) is: 1) The <general value specification> immediately contained in <using argument> shall be either a <host parameter specification> or an <embedded variable specification>. and SQL/Foundation, Subclause 19.11, "<output using clause>", Syntax Rule 1) is: 1) The <target specification> immediately contained in <into argument> shall be either a <host parameter specification> or an <embedded variable specification>. Without these being modified by SQL/PSM, it is thus not currently possible for an SQL parameter to be either a <using argument> or an <into argument>. **Solution** None provided with comment.	
SQL/MED						
	NLD-P09-001		2-Minor Technical	*P09-24.10, ROUTINE_MAPPING_OPTIONS view*	MED-067 The following Possible Problem has been noted: **Source:** DCOR/2004, WG3-P09-003 **Possible Problem:** The View ROUTINE_MAPPING_OPTIONS has no privilege check and no restriction to the actual catalog. **Solution** None provided with comment.	
	NLD-P09-002		2-Minor Technical	*P09-24.11, ROUTINE_MAPPINGS view*	MED-068 The following Possible Problem has been noted: **Source:** DCOR/2004, WG3-P09-004 **Possible Problem:** The View ROUTINE_MAPPINGS has no privilege check and no restriction to	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					the actual catalog. **Solution** None provided with comment.	
	NLD-P09-003		2-Minor Technical	*P09-24.12, USER_MAPPING_OPTIONS view*	MED-069 The following Possible Problem has been noted: **Source:** DCOR/2004, WG3-P09-005 **Possible Problem:** The View USER_MAPPINGS_OPTIONS has no privilege check and no restriction to the actual catalog. **Solution** None provided with comment.	
	NLD-P09-004		2-Minor Technical	*P09-24.13, USER_MAPPINGS view*	MED-070 The following Possible Problem has been noted: **Source:** DCOR/2004, WG3-P09-006 **Possible Problem:** The View USER_MAPPINGS has no privilege check and no restriction to the actual catalog. **Solution** None provided with comment.	
	NLD-P09-005		2-Minor Technical	*P09-25.02, DATA_TYPE_DESCRIPTOR base table*	MED-071 The following Possible Problem has been noted: **Source:** DCOR/2004, WG3-P09-008 **Possible Problem:** The Constraint DATA_TYPE_DESCRIPTOR_DATA_TYPE_CHECK_COMBINATIONS of the table DATA_TYPE_DESCRIPTOR is out of synch with its definition in Part 11 (Schemata). **Solution** None provided with comment.	
	NLD-P09-006		2-Minor Technical	*P09–25.10, ROUTINE_MAPPINGS base table*	MED-072 The following Possible Problem has been noted: **Source:** DCOR/2004, WG3-P09-009 **Possible Problem:** The constraint ROUTINE_MAPPINGS_PRIMARY_KEY requires that the value of the column ROUTINE_MAPPING_NAME is unique across all catalogs in a given DEFINITION_SCHEMA. This seems not be reasonable. **Solution** None provided with comment.	
	NLD-P09-007		3-Major Editorial	*P09-06.02, <cast specification>*	MED-065 The following Possible Problem has been noted: **Source:** FCD1/2002, USA-P09-041 **Possible Problem:** The table in SR 2) is an inappropriate way to add new data types to the casting table in ISO/IEC 9075-2. A different approach would be preferable to avoid problems caused by adding data types in multiple incremental parts (e.g., DATALINK in SQL/MED and	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					XML in SQL/XML). A better approach would be to use a new SR 2) that says something like "Add a new rightmost column to the table following SR 6) in ISO/IEC 9075-2", followed by a table that looks something like this: <data type> *SD* of <data type> of *TD* <value expression> DL EN N AN N ... RW M Then another new SR would be specified, something like this: "Add a new row at the end of the table following SR 6) in ISO/IEC 9075-2" <data type> *SD* of <data type> of *TD* <value expression> EN AN VC FC D T TS YM DT BO UDT CL BL RT CT RW DL DL N N N N N N N N N N N N N N N Y This approach has the advantage of correctly inserting a column and a row, rather than replacing the entire table. However, it leaves the disadvantage that insertion of a column and a row by SQL/MED and another by SQL/XML causes two cells of the table to be unspecified...the cell concerning casting of the data type added by SQL/MED to and from the data type added by SQL/XML. That disadvantage might be resolved by adding (e.g., in Foundation) a statement that such "unspecified cells" are implicitly filled with "N", so that no such casting is supported. **Solution** None provided with comment.	
	NLD-P09-008		3-Major Editorial	*P09-No specific location*	MED-064 The following Possible Problem has been noted: **Source:** FCD1/2002, DEU-P09-980 **Possible Problem:** A look at Clause 4, "Concepts", and associated Subclauses seems to suggest that many columns defined in Clause 25, "Definition Schema", that are presently optional (meaning that a value of null is permitted) should be mandatory. Thus, a careful examination of all column definitions is required, and some of them may require NOT NULL constraints to be added. **Solution** None provided with comment.	
	NLD-P09-009		1-Major Technical	*P09-No specific location*	When the General Rules of another Subclause are invoked the specification of the paramater passing is not always correct. Soemtimes the name of the argument(s) is(are) not explicitly given and sometimes the arguments are not correctly identified. In the latter case is is	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					sometimes because the invoked Subclause does not itself given clearly identifiable names to its arguments. All the calling and called Subclauses should be checked and corrected. **Solution** None provided with comment.	
	NLD-P09-010		2-Minor Technical	*P09-No specific location*	MED-028 The following Language Opportunity has been noted: **Source:** WG3:RTM-017R3/X3H2-99-255R2, Comment WG3-P09-005 **Language Opportunity:** Acceptance of WG3:YGJ-082 made it prohibited to link a single external file more than once. This has been identified as an undesirable restriction in at least some situations. **Solution** None provided with comment.	
	NLD-P09-011		2-Minor Technical	*P09-No specific location*	MED-033 The following Language Opportunity has been noted: **Source:** WG3:RTM-017R3/X3H2-99-255R2, Comment WG3-P09-011 **Language Opportunity:** It is desirable to provide the capability to deal with character sets and collations for character string columns of foreign tables. **Solution** None provided with comment.	
	NLD-P09-012		2-Minor Technical	*P09-No specific location*	MED-045 The following Language Opportunity has been noted: **Source:** WG3:BHX-148/H2-2000-___ **Language Opportunity:** WG3:BHX-148R1 proposed the use of only UTF-16 to communicate character strings between the SQLserver and the foreign-data wrapper. This limitation could profitably be relaxed to permit UTF-8 and/or others, including implementation-defined character sets. **Solution** None provided with comment.	
	NLD-P09-013		2-Minor Technical	*P09-No specific location*	MED-046 The following Language Opportunity has been noted: **Source:** WG3:BHX-108R1/H2-2000-___ and FCD1 2000, GBR-P09-041 **Language Opportunity:** Generic options — some requirements are obvious and should be standardized — for example the name by which the *FT* is known at the *FS* may be different from that in the SQL Environment. If the server is SQL-aware, then the foreign table could be defined by a <query specification>. There is a need for discussion of the costs/benefits/opportunities/mechanisms for further standardization. **Solution** None provided with comment.	
	NLD-P09-014		2-Minor Technical	*P09-No specific location*	MED-047 The following Language Opportunity has been noted: **Source:** FCD1 2000, GBR-P09-043, FCD1 2000, GBR-P09-001, and FCD1	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					2000, GBR-P09-002 **Language Opportunity:** There is a need to acknowledge current implementations: Make FOREIGN DATA WRAPPER optional and add options USE INTERFACE <name> and USE PROTOCOL <name> Use of Standard Interfaces. Where standard interfaces already exist for accessing foreign data, it should be possible to reference the interfaces without requiring Wrappers. Example: Let A and B be RDBMS Vendors; Let X and Y be video specialists. If AX is an implementation of Video using SQL MED and a wrapper WX designed by X and BY is an implementation of Video using SQL MED and a wrapper WY designed by Y then SQL MED does not guarantee that the WY wrapper will work with A or that WX will work with B or that a user of AX can easily port their application to BY. Suppose both X and Y support a standard interface VAPI, then it would be possible to write wrappers that map to VAPI. This might achieve some ability to change video suppliers, but only if the wrapper writers use the VAPI interface with portability in mind. Actual interchangeability is most likely if the wrappers are written by the vendors A and B and supported by them. But in this case the SQL-MED interface becomes an internal one of no interest to users. Use of Protocols. Where foreign data is remote and protocols exist for accessing the remote servers, it should be possible to reference the protocols without requiring wrappers. Example: Let A and B be RDBMS vendors; Let AP and BP be protocols used for accessing remote servers by A and B. Most vendors have a proprietary protocol and many have also implemented their competitors' protocols too. Hence there is already a well defined means of accessing remote data. If these protocols are implemented through wrappers then interchangeability of components could be achieved at three levels: — SQL-MED — A protocol API — The protocol itself Of these, the SQL-MED interface is the most complex, the latest to appear and the most incomplete. It seems to add no value. We think it would be more appropriate to let the foreign server supporting the foreign tables be directly associated with the Protocol **Solution** None provided with comment.	
	NLD-P09-015		2-Minor	*P09-No specific*	MED-055 The following Language Opportunity has been noted:	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
			Technical	*location*	**Source:** FCD1 2000, AUS-P09-007 **Language Opportunity:** There are a number of places in the 'Sequence of actions during foreign server request executions' where the same routine may be called multiple times to return information about options etc. In addition there are some places where Multiple routines are called each returning one item of information at a time from about the particular object. Each of these calls requires a 'context' switch in most operating systems which may in some circumstances end up incurring a substantial operating system overhead in terms of CPU etc. Thus it would be preferable if there were additional alternative methods by which this information could be passed between the SQL Server and the foreign wrapper routines. One mechanism may be to use a structure for various components that may be passed directly to the wrapper routine. Alternatively more than one item of information may be returned by a single call Thus for example, in addition to the following routines — GetServerName — GetServerType — GetServerVersion a single routine GetServerInfo may return all the information. Or in the case where multiple calls would be made to a single routine (for example GetWrapperOption) to return multiple options either an array or a formatted XML document may be used so that a single call may return multiple options. We would like to see some discussion on the possibility of adding optimal multi-return-value procedures to reduce the possible overhead of excessive multiple procedural calls. **Solution** None provided with comment.	
	NLD-P09-016		2-Minor Technical	*P09-No specific location*	MED-056 The following Language Opportunity has been noted: **Source:** FCD1 2000, AUS-P09-007 **Language Opportunity:** The SQLSTATE corresponding to *FDW-specific condition — unable to create reply* is not sufficiently precise or informative. More specific diagnostic information is required. **Solution** None provided with comment.	
	NLD-P09-017		2-Minor Technical	*P09-No specific location*	MED-061 The following Language Opportunity has been noted: **Source:** WG3:YYJ-016 (USA-P09-018) **Language Opportunity:** MED's facility for communicating between the "local" SQL-server and the	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					foreign-data wrapper can be significantly enhanced by providing the ability to pass pre-parsed SQL statements or fragments of them from the SQL-server to the foreign-data wrapper. The most obvious choice for representing this information is in an XML format of some sort, preferably a parse tree or analog. **Solution** None provided with comment.	
	NLD-P09-018		2-Minor Technical	*P09-No specific location*	MED-066 The following Language Opportunity has been noted: **Source:** WG3:DRS-119 **Language Opportunity:** SQL/MED currently only provides read-only access for foreign tables. However, there are applications which require the ability to update data stored in those tables, this includes the ability to create new data and to delete existing data (UID - update, insert, delete). If and when this LO is addressed, changes applied to different foreign tables (possibly residing on different foreign servers) need to be handled according to ACID principles (atomicity, consistency, isolation, durability). Additionally, Subclauses are needed along the lines of those in Foundation, headed "Effect of inserting/ replacing/ deleting...", plus extensions to existing DML Subclauses in Foundation that will cause these new Subclauses to be invoked when appropriate, to handle UID operations correctly. Furthermore, the underlying foreign-data wrapper interface needs to be enhanced to enable UID. It might also be desirable to be able to specify constraints as well as triggers on foreign tables. **Solution** None provided with comment.	
SQL/OLB						
	NLD-P10-001		2-Minor Technical	*P10-09.09 EntryInfo overview*	OLB-002 The following Language Opportunity has been noted: **Source:** First FCD ballot, comment CAN-P10-017 **Language Opportunity:** The exact set of class of statements that Table 3, “Association of roles with SQLJ <executable clause>s” refers to could be more explicitly defined. **Solution** None provided with comment.	
	NLD-P10-002		2-Minor Technical	*P10-No specific location*	OLB-003 The following Language Opportunity has been noted: **Source:** First FCD ballot, comments CAN-P10-023, CAN-P10-024, and CAN-P10-025 **Language Opportunity:** There may be many opportunities to replace D&Rs in SQL/OLB with an informative Note that says something like "Conformance to SQL/OLB requires	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					support only for the ... keywords", when referencing statements or other syntax defined in Foundation or other parts. **Solution** None provided with comment.	
	NLD-P10-003		2-Minor Technical	*P10-No specific location*	OLB-004 The following Language Opportunity has been noted: **Source:** First FCD ballot, comment CAN-P10-026, reinstated by WG3:ZSH-047 = H2-2003-028 **Language Opportunity:** SQL/OLB could benefit from supporting the optional LOCAL keyword in SQL:1999's <set transaction statement>. **Solution** None provided with comment.	
	NLD-P10-004		2-Minor Technical	*P10-09.07.03, Profile customizer interface*	OLB-009 The following Language Opportunity has been noted: **Source:** First FCD ballot, comment DEU-P10-014 **Language Opportunity:** Something needs to be said about how the operations "acceptsConnention" in this subclause and in subclause 5.6, 'Customization interface', relate to each other. **Solution** None provided with comment.	
	NLD-P10-005		2-Minor Technical	*P10-09.09, EntryInfo overview*	OLB-010 The following Language Opportunity has been noted: **Source:** First FCD ballot, comment CAN-P10-018 and WG3:PER-098R1/H2-2001-059 **Language Opportunity:** Table 4, "SQLJ type properties", must be extended to support the new SQL-99 data types (e.g., ARRAY, MULTISET, and ROW). Support for ARRAY has been provided by WG3:DRS-080/H2-2002-458. It is not anticipated that support for either MULTISET or ROW will be provided by SQL/OLB until such time as JDBC provides such support. **Solution** None provided with comment.	
	NLD-P10-006		2-Minor Technical	*P10-09.09, EntryInfo overview*	OLB-011 The following Language Opportunity has been noted: **Source:** First FCD ballot, comment CAN-P10-019 **Language Opportunity:** Table 4, "SQLJ type properties", must be extended to support the SQL-92 data types not mentioned (e.g. DECIMAL, BIT, BIT VARYING, and INTERVAL). Support for DECIMAL is provided via the java.sql.Types values NUMERIC and DECIMAL. Further, per SQL/Foundaiton, Annex E, "Incompatibilities with ISO/IEC 9075-2:2003", ISO/IEC 9075-2:1999 defined data types BIT and BIT VARYING, but those types have been deleted from this edition of ISO/IEC 9075. It is not anticipated that support for INTERVAL will be provided by	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					SQL/OLB until such time as JDBC provides such support. **Solution** None provided with comment.	
	NLD-P10-007		2-Minor Technical	*P10-No specific location*	OLB-014 The following Language Opportunity has been noted: **Source:** First FCD ballot, comment DEU-P10-015 **Language Opportunity:** **Reference:** P10, SQL/OLB, 06.01, Grammar notation (which no longer exists!) This subclause contains conventional material that has traditionally been provided in other parts of 9075 as a subclause of Clause 3. In order to avoid a major rewrite, such a Conventions Subclause should be added to this part; it should merely outline where and how the information one would have expected at that clause is actually provided in this part of 9075. **Solution** None provided with comment.	
	NLD-P10-008		2-Minor Technical	*P10-No specific location*	OLB-015 The following Language Opportunity has been noted: **Source:** First FCD ballot, comment GBR-P10-019 **Language Opportunity:** **Reference:** P10, SQL/OLB, 09.05 (no title given) "Binary portability", more properly "portability of intermediate object code representation", is an objective of the originators of the Java language. Clause 5 does not sufficiently distinguish between statements of intent, tutorial matter and concrete specification. The clause should be merged into the general Concepts clause, and should be further revised to clarify the distinction between things that are part of the SQLJava binding and features of those things that are part of Java. **Solution** None provided with comment.	
	NLD-P10-009		2-Minor Technical	*P10-No specific location*	OLB-017 The following Language Opportunity has been noted: **Source:** First FCD ballot, comment USA-P10-025 **Language Opportunity:** This document contains "Definitions and Rules" clauses that sometimes appear analogous to SQL "Syntax Rules" and sometimes like "General Rules". However, unlike "Syntax Rules" and "General Rules" there is no general specification of the effect of violating a "Definition and Rules" nor of when the "Definition and Rules" are validated/performed. The validation time of and the effect of violation of a "Definition and Rules" rule should be specified. **Solution** None provided with comment.	
	NLD-P10-010		2-Minor Technical	*P10-No specific location*	OLB-018 The following Language Opportunity has been noted: **Source:** First FCD ballot, comment DEU-P10-020 **Language Opportunity:**	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					To improve readability, more cross-references are needed. E.g., when the interfaces are specified that are implemented by some class definition (see for instance 10.2.1) it would be helpful to have reference to the subclause defining that interface. The author of this comment is aware that there is abundant precedence for such cross-references in the document (see "See also" sections). **Solution** None provided with comment.	
	NLD-P10-011		2-Minor Technical	*P10-No specific location*	OLB-025 The following Language Opportunity has been noted: **Source:** Email from Fred Zemke, 2001-11-05, from unknown source **Language Opportunity:** There are many paragraphs that say "An SQLException will be thrown" without saying what that condition is! Is the implementation free to raise any exception that it feels like, possibly even one chosen randomly? If not, then the standard must say what condition is thrown! These places are usually accompanied by an editor's note, which should be removed whenever the problem at that location is solved. **Solution** None provided with comment.	
	NLD-P10-012		2-Minor Technical	*P10-04.09, Default connection context*	OLB-028 The following Language Opportunity has been noted: **Source:** WG3:ZSH-153R1 = H2-2002-153R1 **Language Opportunity:** A problem arises because both SQL/OLB and SQL/JRT have mechanisms for referencing their default SQL-environment. In SQL/OLB, the JNDI registered "jdbc/defaultDataSource" name will, if present, identify the default data source for SQL operations to be performed against. In SQL/JRT, the JDBC URL "jdbc:default:connection" identifies a JDBC connection to the current SQLimplementation, SQL-session, and SQL-transaction. This raises the question: When, if ever, are the following logically equivalent? 1) Connection con = DriverManager.getConnection("jdbc:default:connection"); 2) Connection con = sqlj.runtime.ref.DefaultContext.getDefaultContext().getConnection(); Connection(); 4) Context ctx = new InitialContext(); DataSource ds = (DataSource) ctx.lookup("jdbc/defaultDataSource"); Connection con = ds.getConnection(); That is, when is the java.sql.Connection con, appearing in the above code sequences providing a JDBC connection to the same SQL-implementation? We believe it is desirable, if not required, for an application to be able to run either inside a database as a "stored procedure" or outside as a regular application without having to be recoded, so we ask that above be issue be clarified. **Solution** None provided with comment.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
	NLD-P10-013		2-Minor Technical	*P10-No specific location*	OLB-029 The following Language Opportunity has been noted: **Source:** WG3:ZSH-153R1 = H2-2002-153R1 **Language Opportunity:** SQL/OLB should make it possible for an SQL/OLB application to use the JDBC 3.0 support of what JDBC 3.0 refers to as 'Auto Generated Keys' without having to use JDBC to do so. This capability is often used to access what many DBMSs refer to as a Row ID of a just inserted or updated row. And while SQL/Foundation doesn't standardize a Row ID, the facility would have utility by allowing access to what SQL/Foundation refers to as 'Identity columns' or 'Generated columns'. **Solution** None provided with comment.	
	NLD-P10-014		2-Minor Technical	*P10-No specific location*	OLB-030 The following Language Opportunity has been noted: **Source:** Email from Mark Ashworth, 2004-07-22, SIA Action Item (see minutes for SIA-025) **Language Opportunity:** WG3:SIA-025R1, "A Shorthand for Getting ALL Diagnostics" proposes to add a new diagnostics option to embedded SQL but ignores the question of whether it should also be added to CLI or the SQLJ (SQL/OLB) binding. **Solution** None provided with comment.	
SQL/Schemata						

	NLD-P11-001		1-Major Technical	*P11-05.09, APPLICABLE_ROLES view*	SCHEM-029 The following Possible Problem has been noted:
					Source: WG3: SIA-026r3
					Possible Problem:
					The function and definition of the Information Schema view APPLICABLE_ROLES are given in Subclause 5.9, “APPLICABLE_ROLES view”:
					Function
					Identifies the applicable roles for the current user.
					Definition
					`CREATE RECURSIVE VIEW APPLICABLE_ROLES ( GRANTEE,`
					`ROLE NAME,`
					`IS GRANTABLE ) AS`
					`( ( SELECT GRANTEE, ROLE_NAME, IS_GRANTABLE`
					`FROM DEFINITION SCHEMA.ROLE AUTHORIZATION DESCRIPTORS`
					`WHERE ( GRANTEE IN`
					`( CURRENT USER, 'PUBLIC' )`
					`OR`
					`GRANTEE IN`
					`( SELECT ROLE NAME`
					`FROM ENABLED ROLES ) ) )`
					`UNION`	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					(SELECT RAD.GRANTEE,RAD.ROLE NAME,RAD.IS GRANTABLE FROM DEFINITION SCHEMA.ROLE AUTHORIZATION DESCRIPTORS RAD JOIN APPLICABLE ROLES R ON RAD.GRANTEE = R.ROLE NAME)); The text shown underlined is redundant. It was proposed by ICN-039 as a replacement for the CURRENT_ROLE that had previously been the second element of the first IN list, having been proposed -erroneously, we believe - by PER-193. Before PER-193, that IN list was merely "(CURRENT_USER, 'PUBLIC')", which was consistent with the stated Function of Subclause 5.9, “APPLICABLE_ROLES view”, "Identifies the applicable roles for the current user". Of course the current role, if there is one, is a role that is applicable for the current user, if there is one. In Part 2 SQL/Foundation, Subclause 18.3, “<set role statement>”, GR4) makes sure of that (and in fact applies, possibly erroneously, an even stronger condition). It appears, then, that the text shown underlined should be deleted. However, we hesitate to propose that because we are uncertain as to the real purpose of the APPLICABLE_ROLES view, considering that there isn't always a current user. What roles, if any, are deemed to be applicable, "for" what, when the top cell of the authorization stack of the current SQL-session contains a role name and no user identifier? Is that role name included in the answer? **Solution** None provided with comment.	
	NLD-P11-002		2-Minor Technical	*P11-06.21, DATA_TYPE_DESCRIPTOR base table*	SCHEM-033 The following Possible Problem has been noted: **Source:** WG3:STX-050 Comment WG3-P11-023 **Possible Problem:** There is no foreign key check for the columns SCOPE_CATALOG, SCOPE_SCHEMA, and SCOPE_NAME to the tables table. It is not clear to the Author, if this reference has to be in the same CATALOG or not **Solution** None provided with comment.	
	NLD-P11-003		2-Minor Technical	*P11-06.41, SCHEMATA base table*	SCHEM-031 The following Possible Problem has been noted: **Source:** WG3:STX-050 Comment WG3-P11-020 **Possible Problem:** A foreign key between the table SCHEMATA and the table CHARACTER_SETS is missing. It is not clear to the author, if this Character Set has to reside in the same catalog. If this is the case, the following constraint would resolve the problem: CONSTRAINT SCHEMATA_FOREIGN_KEY_CHARACTER_SETS FOREIGN KEY (DEFAULT_CHARACTER_SET_CATALOG,	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					DEFAULT_CHARACTER_SET_SCHEMA, DEFAULT_CHARACTER_SET_NAME) REFERENCES CHARACTER_SETS **Solution** None provided with comment.	
	NLD-P11-004		2-Minor Technical	*P11–06.62, USER_DEFINED_TYPES base table*	SCHEM-032 The following Possible Problem has been noted: **Source:** WG3:STX-050 Comment WG3-P11-021 **Possible Problem:** The value list and the select list of the last query of the constraint USER_DEFINED_TYPES_CHECK_SOURCE_TYPE do not match. They have different number of elements. It reads: (USER_DEFINED_TYPE_CATALOG, USER_DEFINED_TYPE_SCHEMA, USER_DEFINED_TYPE_NAME, SOURCE_DTD_IDENTIFIER) IN (SELECT OBJECT_CATALOG, OBJECT_SCHEMA, OBJECT_NAME, OBJECT_TYPE, DTD_IDENTIFIER FROM DATA_TYPE_DESCRIPTOR….. **Solution** None provided with comment.	
	NLD-P11-005		2-Minor Technical	*P11-06.11, CHARACTER_SETS base table*	SCHEM-002 The following Language Opportunity has been noted: **Source:** DCOR 2000, SWE-STC-030 **Language Opportunity:** This base table contains a bare minimum of information. It could be enhanced to indicate relationships among character sets, for example whether the character set is standard, implementation-defined, or userdefined, and what characater set a user-defined charater set is based on. **Solution** None provided with comment.	
	NLD-P11-006		2-Minor Technical	*P11-06.44, SQL_IMPLEMENTATION_INFO base table*	SCHEM-008 The following Language Opportunity has been noted: **Source:** WG3:PER-118/H2-2001-??? **Language Opportunity:** Subclause 6.44, “SQL_IMPLEMENTATION_INFO base table”, is defined to contain information about SQL-implementation information items (identified by name and number) but these items are not defined in the other parts of the standard. **Solution** None provided with comment.	
	NLD-P11-007		2-Minor Technical	*P11-06.46, SQL_SIZING base table*	SCHEM-009 The following Language Opportunity has been noted: **Source:** WG3:PER-118/H2-2001-??? **Language Opportunity:** Subclause 6.46, “SQL_SIZING base table”, is defined to contain information about SQL sizing items (identified by name and number) but these items are not	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					defined in the other parts of the standard. (Subclause 6.44, "SQL_IMPLEMENTATION_INFO base table", has the same problem.). **Solution** None provided with comment.	
	NLD-P11-008		2-Minor Technical	*P11-06.21, DATA_TYPE_DESCRIPTOR base table*	SCHEM-013 The following Language Opportunity has been noted: **Source:** WG3:PER-100R2/H2-2001-062R2 **Language Opportunity:** Paper WG3:PER-100r2 noted the following Language Opportunity: The user may wish to recover the original type declaration, rather than the equivalent type declaration that is used by the SQL-server. This concern could be met by adding columns such as ORIGINAL_DATA_TYPE and ORIGINAL_PRECISION to the DATA_TYPE_DESCRIPTORS base table, as well as all views that draw from it. These new columns should be part of a new conformance feature, to make them optional, since not every implementation will be able to display the original type declaration. **Solution** None provided with comment.	
	NLD-P11-009		2-Minor Technical	*P11-06.21, DATA_TYPE_DESCRIPTOR base table*	SCHEM-014 The following Language Opportunity has been noted: **Source:** WG3:PER-100R2/H2-2001-062R2 **Language Opportunity:** Paper WG3:PER-100r2 noted the following Language Opportunity: Users might be interested to know the largest and smallest exponents accomodated by the approximate numeric types. **Solution** None provided with comment.	
	NLD-P11-010		2-Minor Technical	*P11-06.21, DATA_TYPE_DESCRIPTOR base table*	SCHEM-015 The following Language Opportunity has been noted: **Source:** WG3:PER-100R2/H2-2001-062R2 **Language Opportunity:** Paper WG3:PER-100r2 noted the following Language Opportunity: A capabililty would be a table that simply listed all the data type equivalences of the numeric data types. **Solution** None provided with comment.	
	NLD-P11-011		2-Minor Technical	*P11-05.11, ATTRIBUTES view*	SCHEM-018 The following Language Opportunity has been noted: **Source:** DCOR/2002 USA-STC-048 and WG3:ZSH-153R1 = H2-2002-153R1 **Language Opportunity:** The function statement says that this view lists the attributes of structured types that the user has access to. "Access" is ambiguous. As the view is currently defined it appears to mean, "has USAGE or UNDER privilege on". This could be solved by changing "that are accessible" to "that the user has USAGE or UNDER privilege for". However this comment will not suggest that solution.	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Instead, this comment will point out that there are ways to define access to an attribute other than USAGE or UNDER privilege on the attribute's type. First, there are other ways to access the type than through USAGE privilege. The type might be the parameter type of an SQL-invoked routine that the user can execute, it might be the return type of a regular function or method that the user can execute, it might be the type of a column that the user can SELECT, or the type of a selectable nested site such as the field of a row, the element type of a collection, or the attribute type of a different structured type. All of these constitute “access” to a structured type. An analogy can be drawn between user-defined types and domains. Note that the DOMAINS view shows not just the domains that the user has USAGE privilege on; it also shows domains that are the types of columns that the user can access. After defining accessible types, you have the question of what makes an attribute accessible. Is it EXECTUE privilege on the observer? Or perhaps EXECUTE on either the observer or the mutator? Or some other criterion? **Solution** None provided with comment.	
	NLD-P11-012		2-Minor Technical	*P11-05.73, USER_DEFINED_TYPES view*	SCHEM-019 The following Language Opportunity has been noted: **Source:** DCOR/2002 USA-STC-059 and WG3:ZSH-153R1 = H2-2002-153R1 **Language Opportunity:** The word "accessible" in the function is ambiguous. What is meant is those user-defined types for which the user has USAGE or UNDER privilege. However, it is questioned in a separate comment on the ATTRIBUTES view whether "accessible" should be limited to types with USAGE or UNDER privilege. Note that DIRECT_SUPERTYPES view will reveal type T's existence if T is the direct supertype of a type T2 for which the user has USAGE or UNDER privilege, even if the user does not have USAGE or UNDER privilege on T itself. This seems inconsistent. Also, COLUMNS view will display the type T if there is a column whose type is T. It is suspected, but not verified, that the type will also be visible in other views of the Information Schema, wherever the type of a site is displayed (for example, ATTRIBUTES view, FIELDS view, ROUTINE view, PARAMETERS view). Note that DOMAINS view shows a domain if either the user has USAGE privilege on the domain or the user has SELECT privilege on a column whose type is the domain; this provides a precedent that "accessible" is not limited to "has a privilege on". **Solution** None provided with comment.	
	NLD-P11-013		2-Minor Technical	*P11-05.20, “COLUMN_UDT_USAGE view*	SCHEM-021 The following Language Opportunity has been noted: **Source:** WG3:ZSH-153R1 = H2-2002-153R1 **Language Opportunity:** DCOR comment USA-STC-049 pointed out that the join condition joining	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					DEFINITION_SCHEMA.COLUMNS with DEFINITION_SCHEMA.SCHEMATA assumes that the former table has columns named USER_DEFINED_TYPE_CATALOG and USER_DEFINED_TYPE_SCHEMA, which it does not. That comment goes on to suggest that perhaps the intent was to join in DATA_TYPE_DESCRIPTORS, which does. However, if the suggestion in USA-STC-049 were followed, this would not really be sufficient to find all columns that are dependent on a user-defined type. What about columns that are row types with a field that is a user-defined type? Or collection types with an element type that is a user-defined type? See the notion of usage-dependent added to Foundation by WG3:YYJ-083r1. Note that in that paper, it is argued that the notion of usage-dependency does not need to recurse through attributes of a structured type. While this argument is sufficient for the purpose of enforcing RESTRICT or CASCADE semantics, and justifiable for Access Rule checking, does it make sense for this view? For example, if type T1 has an attribute of type T2, and column C1 is of type T1, does C1 depend on T2 in the meaning of this view? If the user is using the view to find all columns to drop before dropping type T2, then the user wants to see C1 in this view. The alternative is that the user must do his own recursion (find all UDTs that depend on T2, then find all columns that depend on any of them.) **Solution** None provided with comment.	
	NLD-P11-014		2-Minor Technical	*P11-06.44, SQL_IMPLEMENTATION_INFO base table*	SCHEM-022 The following Language Opportunity has been noted: **Source:** WG3:ZSH-153R1 = H2-2002-153R1 **Language Opportunity:** There is no list of values for IMPLEMENTATION_INFO_ID and IMPLEMENTATION_INFO_NAME. It seems that many of these were intended to be the codes used in CLI by GetInfo (see for example CLI GetInfo GR 10) subrules b), c), p) and q).) The writer of this comment does not know if there are codes that are necessary for CLI or other parts of SQL. But see CLI subclause 7.1 SQL_IMPLEMENTATION_INFO base table. **Solution** None provided with comment.	
	NLD-P11-015		2-Minor Technical	*P11-No specific location*	SCHEM-023 The following Language Opportunity has been noted: **Source:** WG3:ZSH-153R1 = H2-2002-153R1 **Language Opportunity:** Implementations should not be required to expose columns about optional features that they don't support. For example, in Subclause 5.22, "COLUMNS view", the IS_SELF_REFERENCING column is meaningful only if Feature S051, "Create tables of type", is implemented. If conformance to that feature is not claimed,	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					then references to the column should be prohibited. **Solution** None provided with comment.	
	NLD-P11-016		2-Minor Technical	*P11-05, Information Schema*	SCHEM-026 The following Language Opportunity has been noted: **Source:** DBL:LGW-152/X3H2-97-352 (also DBL:LGW-023/X3H2-97-044, SEQ# 406, USA-102*) **Language Opportunity:** The ROUTINES view and base table have columns that contain the timestamp of when the routine was CREATED and LAST_ALTERED. These are analogous to the file creation and modification timestamps typically provided by a file system. These timestamps are useful for comparing the creation and modification timestamps of the database objects with the timestamps in an external source code control and configuration management utility. Since SQL3 supports extensive programmatic capabilities this configuration management support is extremely useful. However it does not go far enough. Created and Last_altered timestamps would also be useful in the following base tables and their associated views: — ABSTRACT_DATA_TYPES — DOMAINS — TABLES — VIEWS — COLUMNS — ASSERTIONS — CHARACTER_SETS — COLLATIONS — TRANSLATIONS — TRIGGERS — SUB_TABLES **Solution** None provided with comment.	
	NLD-P11-017		2-Minor Technical	*P11-05, Information Schema*	SCHEM-027 The following Language Opportunity has been noted: **Source:** DBL:LGW-152/X3H2-97-352 (also DBL:LGW-023/X3H2-97-044, SEQ#409, USA-105) **Language Opportunity:** Many "information discovery" products depend upon full text searches of document databases to feed the indexing mechanisms used in their search engines. It is very difficult to extend this technique to "structured" relational databases especially if they have high numeric content unless there is some textual description of the semantics associated with data values and schema objects. Sometimes "information discovery" agents will search the INFORMATION_SCHEMA Catalog Schema Table and	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					Column names looking for relevant key word "stems" to feed to the search engine. It would be very helpful to users of such agents if there were a "standard" way to read and write textual descriptions of what each schema object represents. Certainly Information Resource Dictionary Systems (IRDS) could help in this task or users could define a special schema for this purpose but at present there is no dependable standard mechanism to make such information available to outside agents. One easy-to-implement yet very helpful facility would be to associate a "COMMENT" or "DESCRIPTION" column with each relevant table in the INFORMATION_SCHEMA together with a "SET SCHEMA COMMENT statement" (or other appropriate syntax) that would allow the owner of a schema object to set and/or modify the COMMENT column associated with it. The normal Information Schema view definition would then determine which users are able to read the COMMENT column so information discovery agents would be able to "discover" whatever comments exist for PUBLIC schema objects and report back to their creators any interesting database content. In addition to information discovery agents comment or description information is crucial to support the reusability of ADTs. An SQL programmer must know what an ADT is supposed to do in order to correctly utilize or subtype it. This information can only be provided by the ADT creator in a text format and is much more likely to be useful if stored in the INFORMATION_SCHEMA than if stored in paper documentation at the bottom of a stack on someone else's desk. This could be accomplished by adding syntax to the ADT definition to support a large amount of text. The SQL objects for which comment/description information would be useful include: DOMAINS, TABLES, VIEWS, COLUMNS, ASSERTIONS, CHARACTER_SETS, COLLATIONS, TRANSLATIONS, TRIGGERS, SUB_TABLES, as well as distinct types, abstract data types, and SQL-invoked routines. **Solution** None provided with comment.	
	NLD-P11-018		1-Major Technical	*P11-No specific location*	SCHEM-028 The following Language Opportunity has been noted: **Source:** WG3:HBA-034R2 = H2-2003-343R4 **Language Opportunity:** We have at least the following kinds of SQL-schema objects that might be involved in a dependency relationship — check constraints	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					— assertions — generated columns — SQL-invoked routines — triggers — views — character sets — collations — transliterations — domains — non-generated columns — base tables — sequences — user-defined types We do not have Information Schema views to report all possible dependencies between these kinds of SQL-schema objects, as seen in this table: (See LO) In the preceding table, • A blank cell means that the dependency cannot occur. • 'y' means that an Information Schema view exists to report such dependencies. • 'N' means that such dependencies can occur, but there is no reporting mechanism currently. Most of the possible dependencies are explained as follows: • A <value expression> can be a <scalar subquery>, which can be a grouped query, which can depend on a check constraint, assertion, or unique constraint in order to deduce a functional dependency. Thus, anything that permits a <value expression> can be dependent on a check constraint, assertion or unique constraint (but only if Feature T301, “Functional dependencies” is supported). • CAST to a character string type with a user-defined character set implies a dependency on the character set. Thus anything permitting a <value expression> might be dependent on a character set. • A collation can be used in comparison predicates, and thus most kinds of SQL-schema objects might depend on a collation. • A <value expression> can contain a CONVERT expression, which depends on a transliteration, so most kinds of SQL-schema objects might depend on a transliteration. Conversely, a transliteration uses an SQL-invoked routine, so a transliteration can be dependent on anything that a routine can be dependent on. Speaking in orders of magnitude, if we have n kinds of SQL-schema objects, and we add one more, then we have (n + 1)2 - n2 = 2n + 1 new kinds of dependency to think about. Thus the cost of adding one kind of SQL-schema	

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
					object is potentially 2n + 1 new kinds of dependency. Aside from the fact that so many kinds of dependency are currently unsupported, [Fred Zemke thinks] that the technique of creating one Information Schema view for each kind of dependency has become unmanageable for our users, and unmaintainable for ourselves. Therefore, [Fred thinks] it is time to come up with a different model for dependency tracking and reporting. [Fred thinks] the correct approach is to define a base table to track immediate dependencies between all kinds of SQL-schema objects, and a recursive Information Schema view that shows all deducible dependencies. **Solution** None provided with comment.	
	NLD-P11-019		2-Minor Technical	*P11-06, Definition Schema*	SCHEM-030 The following Language Opportunity has been noted: **Source:** WG3:SIA-018 = ANSI NCITS H2-2003-429 / email from Joern Bartels **Language Opportunity:** SIA-018 adds the Subclause 10.11, "Determination of view and view component privileges" to Part 2. This subclause introduces the new view privilege dependency descriptor. There is no corresponding base table in Clause 6, "Definition Schema" of Part 11 defined. As this descriptor is created in the Subclause 12.1, "<grant statement>" of Part 2 and used in the Subclause 12.7, "<revoke statement>" of Part 2, it needs to be stored somewhere. **Solution** None provided with comment.	
SQL/JRT						
	NLD-P13-001		2-Minor Technical	*P13-No specific location*	JRT-001 The following Language Opportunity has been noted: **Source:** WG3:YYJ-041 = H2-2001-405 **Language Opportunity:** Subclause 4.8.5.1, "SERIALIZABLE", should perhaps say "implements `java.io.Serializable` or any Java equivalent". This would also permit, for example, implementing Externalizable, which can often be done with better performance and space usage than Serializable. **Solution** None provided with comment.	

UK Ballot Comments on ISO/IEC 9075-4 CD — 2005-03-09

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
{CD} SQL/PSM						
	GBR-P04-001		4-Minor Editorial	*P04-09-04, <column definition>*	The replacement directive in Syntax Rule 1 should reference Syntax Rule 1 of the corresponding clause in Part 2, rather than Syntax Rule 0.1. It may be worth noting that in replacements of rules for the purpose of inserting one item in a list embedded in the text, it will often be clearer to add a new rule, rather than leaving the reader to work out what has changed. Such an approach would also provide some protection against modification of the same rule by two different Parts. A major editorial opportunity to be exploited only in the unlikely situation of an editor having time on his hands? **Solution** See main comment	
	GBR-P04-002		4-Minor Editorial	*P04-09-07, <drop column scope clause>*	It is not immediately obvious whether the sentence fragment specified as Syntax Rule 1 is to be added at the same level as the lead-in text of SR5 of Subclause 11.16 part 2, or to be added as a new item in the list. The editing instruction should be modified to make it clear that the fragment is to be added as a new entry at the end of the list. **Solution** See comment	

USA Comments on SC32 N 1202: ISO/IEC CD 9075-4

SEQ #	Cmnt ID	See Also	Severity	Reference	Description	Addressed By
	USA-P04-010	USA-P03-010	3-Major Editorial	*P04-02.01, JTC1 standards*	The references for [Framework], [Foundation], and [Schemata] are out-of-date. **Solution** None provided with comment.	

	USA-P04-020		1-Major Technical	*P04-13.13. <for statement>*	Modify Syntax Rule 7) as shown here (see also STX-041):
					7) Let *COMMON_CODE* be:
					`...`
					`BEGIN NOT ATOMIC`
					`OPEN CN;`
					`BL:LOOP`
					~~`OPEN CN,`~~
					`BEGIN NOT ATOMIC`
					`...`
					Solution
					Provided with comment.
						See comment

	USA-P04-999		1-Major Technical	*P04-No specific location*	All Possible Problems and Editor's Notes must be satisfactorily resolved and all problems discovered during the course of the ballot resolution process must be satisfactorily resolved. **Solution** None provided with comment.	

<<< End >>>